

Software

IDC DOCUMENTATION

Threshold Monitoring Subsystem



**Approved for public release;
distribution unlimited**

Notice

This document was published July 2001 by the Monitoring Systems Operation of Science Applications International Corporation (SAIC) as part of the International Data Centre (IDC) Documentation. Every effort was made to ensure that the information in this document was accurate at the time of publication. However, information is subject to change.

Contributors

Lyla Taylor, NORSAR
Tormod Kværna, NORSAR
Frode Ringdal, NORSAR

Trademarks

BEA TUXEDO is a registered trademark of BEA Systems, Inc.
ORACLE is a registered trademark of Oracle Corporation.
SAIC is a trademark of Science Applications International Corporation.
Solaris is a registered trademark of Sun Microsystems.
Sun is a registered trademark of Sun Microsystems.
UNIX is a registered trademark of UNIX System Labs, Inc.

Ordering Information

The ordering number for this document is SAIC-01/3015.

This document is cited within other IDC documents as [IDC7.1.14].

Threshold Monitoring Subsystem

CONTENTS

| | |
|--|-----|
| <u>About this Document</u> | i |
| ■ <u>PURPOSE</u> | ii |
| ■ <u>SCOPE</u> | ii |
| ■ <u>AUDIENCE</u> | ii |
| ■ <u>RELATED INFORMATION</u> | iii |
| ■ <u>USING THIS DOCUMENT</u> | iii |
| <u>Conventions</u> | iv |
| <u>Chapter 1: Overview</u> | 1 |
| ■ <u>INTRODUCTION</u> | 2 |
| ■ <u>FUNCTIONALITY</u> | 4 |
| ■ <u>IDENTIFICATION</u> | 5 |
| ■ <u>STATUS OF DEVELOPMENT</u> | 6 |
| ■ <u>BACKGROUND AND HISTORY</u> | 6 |
| ■ <u>OPERATING ENVIRONMENT</u> | 6 |
| <u>Hardware</u> | 6 |
| <u>Commercial-Off-The-Shelf Software</u> | 7 |
| <u>Chapter 2: Architectural Design</u> | 9 |
| ■ <u>CONCEPTUAL DESIGN</u> | 10 |
| <u>Generating Threshold Traces</u> | 10 |
| <u>Calculating Detection Capability Traces</u> | 12 |
| <u>Calculating Detection Capability Traces for Global Grid Systems</u> | 13 |
| <u>Calibrating Magnitudes and Setting Time/Azimuth/Slowness Tolerances</u> | 15 |
| <u>Setting Station-specific Parameters</u> | 17 |
| ■ <u>DESIGN DECISIONS</u> | 21 |
| <u>Programming Language</u> | 21 |

| | |
|---|----|
| Global Libraries | 22 |
| Filesystem | 22 |
| Web | 22 |
| Design Model | 22 |
| Database Schema Overview | 23 |
| ■ FUNCTIONAL DESCRIPTION | 24 |
| Creating Processing Sessions | 25 |
| Altering Processing Sessions | 26 |
| Generating STA Traces for Each Station | 26 |
| Generating Threshold Traces for Global Grid | 26 |
| Interpolating and Reformatting Threshold Traces for Global Grid | 26 |
| Generating Products | 27 |
| Updating Products with Event Information | 27 |
| ■ INTERFACE DESIGN | 28 |
| Interface with Other IDC Systems | 28 |
| Interface with External Users | 29 |
| Interface with Operators | 29 |
| Chapter 3: Detailed Design | 31 |
| ■ DATA FLOW MODEL | 32 |
| ■ PROCESSING UNITS | 34 |
| CreateTMSession | 34 |
| AddTMStation | 46 |
| DeleteTMStation | 47 |
| TMthreshold | 48 |
| TMmap | 51 |
| TMprod | 54 |
| TMpostreb | 58 |
| LoopCopy | 61 |
| ■ DATABASE DESCRIPTION | 64 |
| Database Design | 64 |
| Database Schema | 64 |

| | |
|-----------------------------------|----|
| <u>References</u> | 67 |
| <u>Glossary</u> | G1 |
| <u>Index</u> | I1 |

Threshold Monitoring Subsystem

FIGURES

| | | |
|-----------------------------------|--|----|
| <u>FIGURE 1.</u> | <u>IDC SOFTWARE CONFIGURATION HIERARCHY</u> | 3 |
| <u>FIGURE 2.</u> | <u>RELATIONSHIP OF TM TO OTHER SOFTWARE UNITS AND CSCIS</u> | 4 |
| <u>FIGURE 3.</u> | <u>REPRESENTATIVE HARDWARE CONFIGURATION FOR TM</u> | 7 |
| <u>FIGURE 4.</u> | <u>2,562 GLOBAL MONITORING TARGET POINTS</u> | 14 |
| <u>FIGURE 5.</u> | <u>ASAR P-PHASE BEAM DEPLOYMENT</u> | 15 |
| <u>FIGURE 6.</u> | <u>MAGNITUDE CALIBRATION FUNCTION AND TRAVEL-TIME CURVE FOR EVENTS AT ZERO DEPTH</u> | 17 |
| <u>FIGURE 7.</u> | <u>ILAR REGIONAL EVENT SIGNAL LOSS</u> | 21 |
| <u>FIGURE 8.</u> | <u>TM FUNCTIONS</u> | 25 |
| <u>FIGURE 9.</u> | <u>TM DATA FLOW MODEL</u> | 33 |
| <u>FIGURE 10.</u> | <u>SESSION DIRECTORY STRUCTURE</u> | 40 |
| <u>FIGURE 11.</u> | <u>TM TABLE RELATIONSHIPS</u> | 65 |

Threshold Monitoring Subsystem

TABLES

| | | |
|-----------------------------------|---|----|
| <u>TABLE I:</u> | <u>DATA FLOW SYMBOLS</u> | iv |
| <u>TABLE II:</u> | <u>ENTITY-RELATIONSHIP SYMBOLS</u> | v |
| <u>TABLE III:</u> | <u>TYPOGRAPHICAL CONVENTIONS</u> | v |
| <u>TABLE IV:</u> | <u>TECHNICAL TERMS</u> | vi |
| <u>TABLE 1:</u> | <u>DATABASE TABLES USED BY TM</u> | 23 |
| <u>TABLE 2:</u> | <u>TM FPDESCRIPTION ROWS</u> | 24 |
| <u>TABLE 3:</u> | <u>STANDARD PRODUCTS AVAILABLE THROUGH TM</u> | 27 |
| <u>TABLE 4:</u> | <u>TERMS RELATING TO STANDARD PRODUCTS</u> | 28 |
| <u>TABLE 5:</u> | <u>CREATETMSESSION PARAMETERS</u> | 35 |
| <u>TABLE 6:</u> | <u>TARGETS.N PARAMETERS</u> | 36 |
| <u>TABLE 7:</u> | <u>BEAMBASIC PARAMETERS</u> | 37 |
| <u>TABLE 8:</u> | <u>MISSTEER.DLOW-DHIG PARAMETERS</u> | 38 |
| <u>TABLE 9:</u> | <u>TM.SITE PARAMETERS</u> | 38 |
| <u>TABLE 10:</u> | <u>TAB.PTM PARAMETERS</u> | 39 |
| <u>TABLE 11:</u> | <u>BEAMS/TMNNN PARAMETERS</u> | 42 |
| <u>TABLE 12:</u> | <u>TM_RECIPES/TARGET_NNNN PARAMETERS</u> | 43 |
| <u>TABLE 13:</u> | <u>TM_RECIPES/TRVMINMAX PARAMETERS</u> | 45 |
| <u>TABLE 14:</u> | <u>TMTHRESHOLD PARAMETERS</u> | 48 |
| <u>TABLE 15:</u> | <u>TMMAP PARAMETERS</u> | 52 |
| <u>TABLE 16:</u> | <u>TMSTATUS.PAR PARAMETERS</u> | 57 |
| <u>TABLE 17:</u> | <u>DATA.CPT PARAMETER FILES</u> | 58 |
| <u>TABLE 18:</u> | <u>LOOPCOPY PARAMETERS</u> | 62 |
| <u>TABLE 19:</u> | <u>DATABASE USAGE BY TM</u> | 64 |

About this Document

This chapter describes the organization and content of the document and includes the following topics:

- [Purpose](#)
- [Scope](#)
- [Audience](#)
- [Related Information](#)
- [Using this Document](#)

About this Document

PURPOSE

This document describes the design and requirements of the Threshold Monitoring (TM) software of the International Data Centre (IDC). The software is part of the Performance Monitoring computer software component (CSC) of the System Monitoring Computer Software Configuration Item (CSCI). This document provides a basis for implementing, supporting, and testing the software.

SCOPE

The Threshold Monitoring software is identified as follows:

Title: Threshold Monitoring

Abbreviation: TM

This document describes the architectural and detailed design of the software including its functionality, components, data structures, high-level interfaces, method of execution, and underlying hardware. This information is modeled on the Data Item Description for *Software Design Descriptions* [\[DOD94a\]](#).

AUDIENCE

This document is intended for all engineering and management staff concerned with the design and requirements of all IDC software in general and of TM in particular. The detailed descriptions are intended for programmers who will be developing, testing, or maintaining TM.

RELATED INFORMATION

The following documents complement this document:

- *Database Schema, Revision 2* [\[IDC5.1.1Rev2\]](#)
- *Distributed Application Control System (DACS) Software User Manual, Revision 0.1* [\[IDC6.5.2Rev0.1\]](#)
- *Threshold Monitoring Software User Manual* [\[IDC6.5.14\]](#)
- *Distributed Application Control System (DACS)* [IDC7.3.1]

See [“References” on page 67](#) for a list of documents that supplement this document. The following UNIX manual (man) pages apply to software used by TM:

- *DFX*
- *tuxshell*

USING THIS DOCUMENT

This document is part of the overall documentation architecture for the IDC. It is part of the Software category, which describes the design of the software. This document is organized as follows:

- [Chapter 1: Overview](#)
This chapter provides a high-level view of TM, including its functionality, components, background, status of development, and current operating environment.
- [Chapter 2: Architectural Design](#)
This chapter describes the architectural design of TM, including its conceptual design, design decisions, functions, and interface design.
- [Chapter 3: Detailed Design](#)
This chapter describes the detailed design of TM including its data flow, software units, and database design.
- [References](#)
This section lists the sources cited in this document.

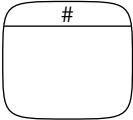

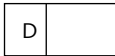
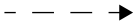

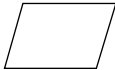
▼ About this Document

- [Glossary](#)
This section defines the terms, abbreviations, and acronyms used in this document.
- [Index](#)
This section lists topics and features provided in the document along with page numbers for reference.

Conventions

This document uses a variety of conventions, which are described in the following tables. [Table I](#) shows the conventions for data flow diagrams. [Table II](#) shows the conventions for entity-relationship diagrams. [Table III](#) lists typographical conventions. [Table IV](#) explains certain technical terms that are not part of the standard Glossary, which is located at the end of this document.

TABLE I: DATA FLOW SYMBOLS

| Description | Symbol ¹ |
|---|---|
| process |  |
| external source or sink of data |  |
| data store D = disk store Db = database store |  |
| control flow |  |
| data flow |  |
| product |  |

1. Symbols in this table are based on Gane-Sarson conventions [\[Gan79\]](#).

TABLE II: ENTITY-RELATIONSHIP SYMBOLS











| Description | Symbol | | | | | | | |
|--|--|------------------|--|--|--------------------|--------------------|-----|--------------------|
| One A maps to one B. | A  B | | | | | | | |
| One A maps to zero or one B. | A  B | | | | | | | |
| One A maps to many Bs. | A  B | | | | | | | |
| One A maps to zero or many Bs. | A  B | | | | | | | |
| database table | <table><tr><td>tablename</td></tr><tr><td> <i>primary key</i></td></tr><tr><td> <i>foreign key</i></td></tr><tr><td><i>attribute 1</i></td></tr><tr><td><i>attribute 2</i></td></tr><tr><td>...</td></tr><tr><td><i>attribute n</i></td></tr></table> | tablename |  <i>primary key</i> |  <i>foreign key</i> | <i>attribute 1</i> | <i>attribute 2</i> | ... | <i>attribute n</i> |
| tablename | | | | | | | | |
|  <i>primary key</i> | | | | | | | | |
|  <i>foreign key</i> | | | | | | | | |
| <i>attribute 1</i> | | | | | | | | |
| <i>attribute 2</i> | | | | | | | | |
| ... | | | | | | | | |
| <i>attribute n</i> | | | | | | | | |

TABLE III: TYPOGRAPHICAL CONVENTIONS

| Element | Font | Example |
|---|----------------|------------------------------------|
| database table | bold | fpdescription |
| database attributes | <i>italics</i> | <i>orid</i> |
| processes, software units, and libraries | | <i>TMthreshold</i> |
| user-defined arguments and variables used in parameter (par) files or program command lines | | CreateTMSession par=<parfile> |
| titles of documents | | <i>Database Schema</i> |
| filenames, directories, and web-sites | courier | <i>staticdir/Targets/targets.n</i> |
| text that should be typed in exactly as shown | | role=TMprod |

▼ About this Document

TABLE IV: TECHNICAL TERMS

| Term | Description |
|--------|---|
| GMT | Generic Mapping Tools [Wes95] |
| IASP91 | name of a particular earth model [Ken91b] |

Chapter 1: Overview

This chapter provides a general overview of the TM software and includes the following topics:

- [Introduction](#)
- [Functionality](#)
- [Identification](#)
- [Status of Development](#)
- [Background and History](#)
- [Operating Environment](#)

Chapter 1: Overview

INTRODUCTION

The software of the IDC acquires time-series and radionuclide data from stations of the International Monitoring System (IMS) and other locations. These data are passed through a number of automatic and interactive analysis stages, which culminate in the estimation of location and in the origin time of events (earthquakes, volcanic eruptions, and so on) in the earth, including its oceans and atmosphere. The results of the analysis are distributed to States Parties and other users by various means. Approximately one million lines of developmental software are spread across six CSCIs of the software architecture. One additional CSCI is devoted to run-time data of the software. [Figure 1](#) shows the logical organization of the IDC software. The System Monitoring CSCI monitors system performance through the following CSCs:

- System Monitoring
This software consists of two scripts that measure the recent input data completeness on individual stations and channels, respectively.
- Performance Monitoring
This software consists of scripts and tools used to assess and report the performance of the system. It includes scripts that run the Fixed Data Set Test.

The TM software is part of the Performance Monitoring CSC. [Figure 2](#) shows the relationship of TM to the Automatic Processing, Interactive Processing, Distributed Processing, and Data Services CSCs.

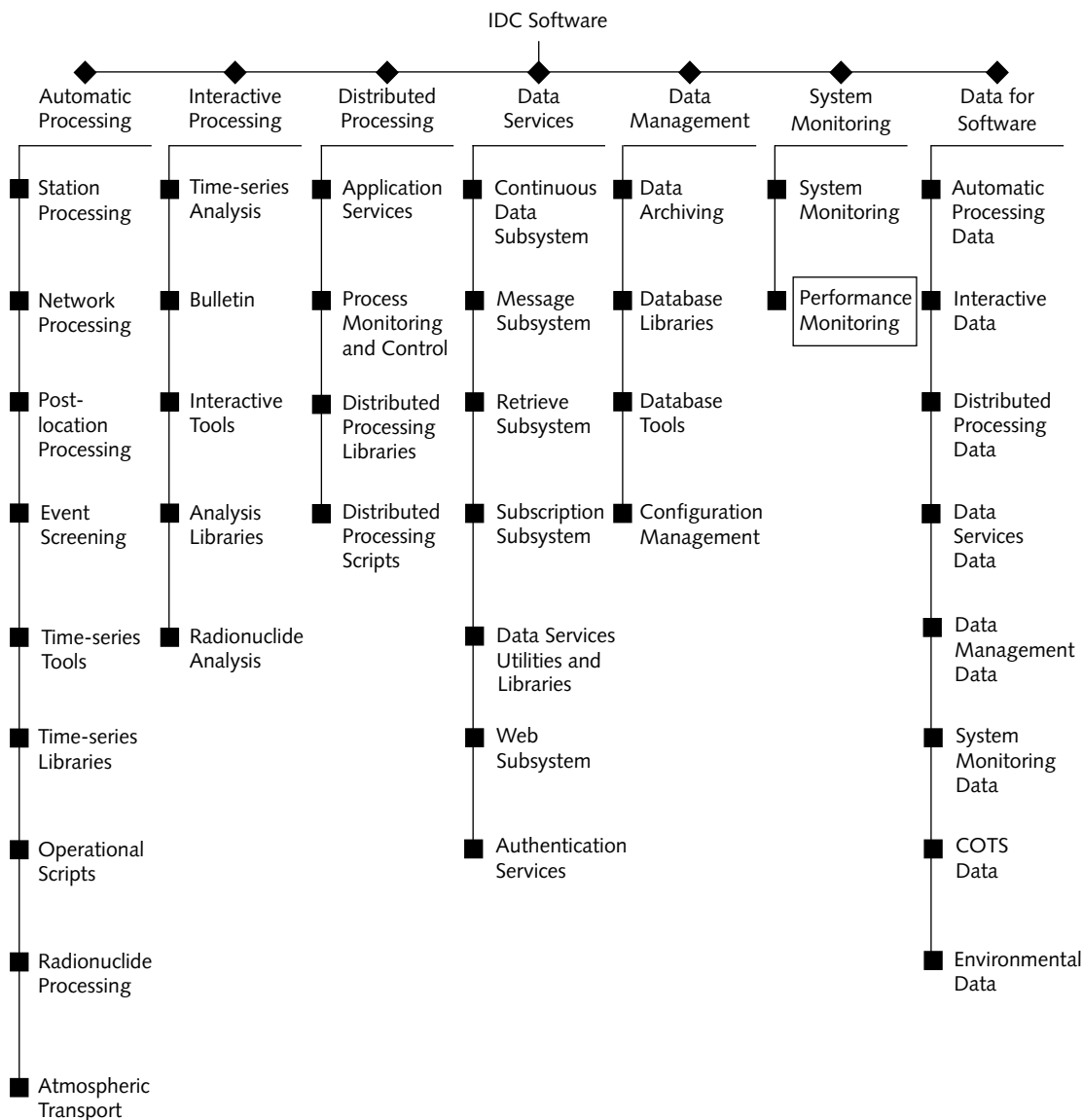


FIGURE 1. IDC SOFTWARE CONFIGURATION HIERARCHY

▼ Overview

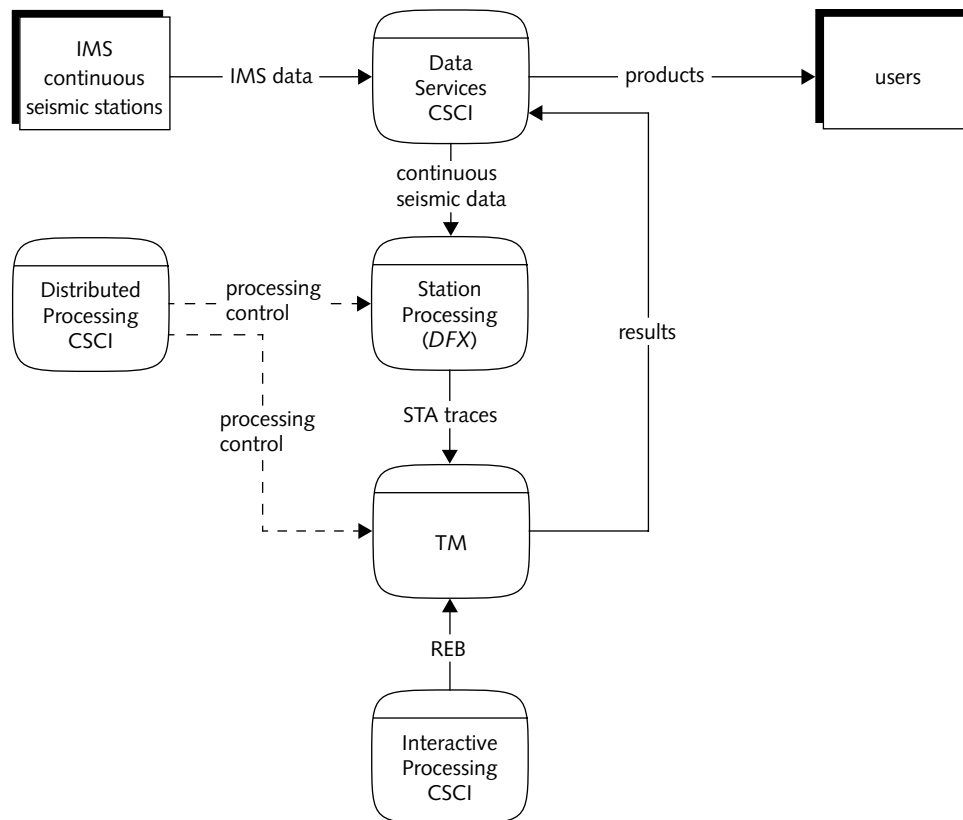


FIGURE 2. RELATIONSHIP OF TM TO OTHER SOFTWARE UNITS AND CSCIS

FUNCTIONALITY

The main function of TM is to assist in monitoring compliance with the Comprehensive Nuclear-Test-Ban Treaty (CTBT), in particular, by placing confidence in the performance of the primary seismic network of the IMS, but also by giving a warning in the case of lowered monitoring capability, for example, caused by station outages, communication problems, data processing problems, or extremely high seismic activity. TM produces three products to accomplish this function: plots of data availability and interfering events for one-hour intervals, an overview of the

background noise levels and the observed signals at each of the primary stations during the data interval, and maps of network detection capability for the data interval.

IDENTIFICATION

TM's components are identified as follows:

- *TMthreshold*
- *TMmap*
- *makecdf.2562*
- *TMprod*
- *tmstatus*
- *tm_stast*
- *plotuptime*
- *detplot*
- *rdf2cdf*
- *TMpostreb*
- *TMbulletin*
- *replotuptime*
- *loopuptime*
- *CreateTMSession*
- *AddTMStation*
- *DeleteTMStation*
- *tm_beambasic*
- *tm_globrec*
- *LoopCopy*
- *CopyPSFile*
- *makelibs*
- *makemods*

STATUS OF DEVELOPMENT

TM was developed by NORSAR and was delivered for use at the IDC in 1998. Version 3.4 of the TM software is described in this document. That version includes a few changes to *CreateTMSession*, *AddTMStation*, *DeleteTMStation*, and *LoopCopy*. One print statement was changed in *TMmap*, and the *ctms* library was updated. Code for formerly “external” libraries is included. The static data directory in version 3.4 contains new station processing recipes for ARCES, NVAR, and PDYAR. Future versions may be released with new station processing recipes.

BACKGROUND AND HISTORY

Tormod Kværna of NORSAR developed the TM software. It was introduced in a paper written by Frode Ringdal and Tormod Kværna in 1989 [\[Rin89\]](#).

TM was implemented in the Prototype International Data Centre (PIDC) at the Center for Monitoring Research (CMR) in Arlington, Virginia, U.S.A. in 1998 and is currently an element of the International Data Centre of the Comprehensive Nuclear-Test-Ban Treaty Organization (CTBTO IDC) in Vienna, Austria.

OPERATING ENVIRONMENT

The following paragraphs describe the hardware and commercial-off-the-shelf (COTS) software required to operate TM.

Hardware

TM runs on a Sun workstation such as the Ultra 10. Typically, the hardware is configured with 128 MB of memory and a minimum of 2.5 GB of magnetic disk. A 4 GB disk should be dedicated to this task to accommodate future expansion of the primary seismic network or any increase in the length of the files written. Due to the amount of input/output (I/O) performed, the disk to which TM will read and write must be mounted locally on the machine on which TM runs; otherwise, the performance of TM will be significantly retarded.

One year's worth of output files from TM requires about 4.7 GB. Sufficient disk space should also be available for long-term storage of these files. Figure 3 shows a representative hardware configuration.

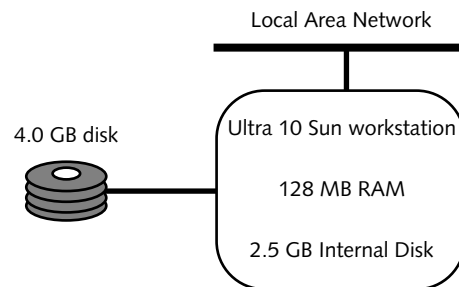


FIGURE 3. REPRESENTATIVE HARDWARE CONFIGURATION FOR TM

Commercial-Off-The-Shelf Software

TM was designed using Solaris 2.6 and ORACLE 7.3.4, but runs using later versions (Solaris 2.7 and ORACLE 8.1.5). The software also depends on the Generic Mapping Tools (GMT 3.3.6 plus errata) software [\[Wes95\]](#) and BEA TUXEDO (version 6.5).

Chapter 2: Architectural Design

This chapter describes the architectural design of TM and includes the following topics:

- [Conceptual Design](#)
- [Design Decisions](#)
- [Functional Description](#)
- [Interface Design](#)

Chapter 2: Architectural Design

CONCEPTUAL DESIGN

Traditional methods for assessing the network detection capability use statistical models for the noise and signal distributions to calculate the detection thresholds as a function of the number of phase detections required for defining an event ([Syk82], [Har85], [Han85], [Rin86], and [Ser89]). The noise models used in these capability assessments are not able to accommodate the effect of interfering signals, such as the coda of large earthquakes, which may cause the estimated thresholds to be quite unrealistic at times. Nor can these methods include effects like communication problems and data processing deficiencies. The TM approach ([Rin89], [Rin92], [Kvæ97a], and [Kvæ97b]) incorporates all of the effects listed above and provides a continuous assessment of the detection capability of the primary seismic network.

Generating Threshold Traces

Following [Rin92], assume a network of N seismic stations ($i = 1, 2, \dots, N$) and M seismic phases ($j = 1, 2, \dots, M$). For a seismic event of magnitude $m_b = m$ an estimate \hat{m}_{ij} of m is given by equation (1):

$$\hat{m}_{ij} = \log S_{ij} + b_j(\Delta, h) \quad (1)$$

where S_{ij} is the measured signal power of the j -th phase at the i -th station and $b_j(\Delta, h)$ is a distance-depth correction term for the j -th phase.

The signal power S_{ij} is usually estimated as A/T , the amplitude of the ground movement (A) divided by the dominant signal period (T). In this case, S_{ij} is assumed to be the measured signal power of the ground velocity (in other words,

the short-term average or STA) at the expected signal arrival time. The signal power is measured on a single channel or array beam filtered in an appropriate frequency band.

Traditionally, equation (1) is defined only for the time window corresponding to a detected seismic event. Consider the right side of equation (1) as a continuous function of time, and define the “threshold parameter” $a_{ij}(t)$ as follows:

$$a_{ij}(t) = \log S_{ij}(t) + b_j(\Delta, h) \quad (2)$$

Equation (2) is the upper magnitude limit for a hypothetical seismic event at a given geographical location (target region) as a function of time. If an actual event does occur at the site, a_{ij} for the time that the signal arrives is the event magnitude estimate for the station. By definition, the function is tied to a specific station and a specific phase.

Using a statistical approach, and assuming statistical independence of the observations [Rin89], a network-based representation of the upper magnitude limit is obtained by considering the function (3):

$$g(m, t) = 1 - \prod_{i,j} \left\{ 1 - \Phi \left(\frac{[m - a_{ij}(t)]}{\sigma_{ij}} \right) \right\} \quad (3)$$

where m is the event magnitude, σ_{ij} is the standard deviation of the assumed magnitude distribution for the i -th station and j -th phase, and Φ denotes the standard (0,1) normal distribution function.

The function $g(m, t)$ is the probability that a given (hypothetical) seismic event of magnitude m at time t would generate signals that exceed the observed noise values for at least one station of the network. For a given t , the function $g(m, t)$ is a monotonically increasing function of m , with values between 0 and 1. A 90 percent upper limit at time t is defined as the solution to equation (3) for which

$$g(m, t) = 0.90 \quad (4)$$

The solution is a function of t , $m_{T90}(t)$, and is called the threshold trace for the network and target region being considered.

Calculating Detection Capability Traces

The threshold trace developed in the previous section is not directly related to the detection capability of the network in the usual sense. Nevertheless, the general method described can be used to obtain a continuous estimate of the n -station network detection capability. To do this, the required signal-to-noise ratio (snr) is added to each individual station threshold trace, and the level is adjusted to correspond to a probability level of 90 percent for phase detection at each station. Let t_{ij} represent the snr detection limit (in log units) for the i -th station and the j -th phase, and let $d_{ij}(t)$ represent the detection threshold (in magnitude units) for that station. Let σ_{ij} represent the assumed standard deviation of the signal. The detection threshold is given by equation (5):

$$d_{ij}(t) = a_{ij}(t) + t_{ij}(t) + 1.28\sigma_{ij} \quad (5)$$

Consider only P-type phases (the extension to the general case is obvious). Eliminating the index j , order the individual station detection thresholds so that:

$$d_1(t) \leq d_2(t) \leq \dots \leq d_N(t) \quad (6)$$

The n -station network detection threshold at time t is defined as the magnitude value $d_n(t)$. For a hypothetical event at this magnitude, at least n stations would be expected to exceed their respective detection thresholds, thus allowing for a network detection of the hypothetical event.

This formulation is quite different from the standard methods for network detection threshold estimation [Har85]. Standard methods assume a statistical distribution of noise and signal amplitude levels and do not utilize the continuously recorded seismic field. Standard methods also employ a somewhat more complicated combinatorial technique, which has been simplified by ordering the individual station thresholds by increasing magnitude.

It would be feasible to calculate the detection thresholds using the combinatorial technique on a continuous basis by using the individual station threshold traces as input. However, examples have shown that the simplified calculation gives results that are generally consistent with combinatorial calculations and that the differences in the results are well below the inherent uncertainties in either method.

Calculating Detection Capability Traces for Global Grid Systems

Global threshold monitoring is achieved by conducting site-specific monitoring of a grid of target points covering the globe. The density of the grid and the interpolation technique applied generally determines the quality of the results.

The method described in [\[Vin92\]](#) was adopted to develop global grid point systems. This method applies triangulation of an icosahedron to construct regularly sampled wavefronts and provides a nearly uniform geographical coverage of the globe at a specified grid density [\[Kvæ92\]](#).

The grid density to be used is mainly a cost-performance trade-off. A 2,562-point grid was chosen for the initial version of TM. [Figure 4](#) shows this grid system in which the region belonging to each grid point corresponds to a circle with a radius of 2.7 degrees. Each grid point is located at the surface of the earth.

The density of the global grid is quite different from the beam deployment density for the arrays in the station network. For each array, a certain number of steering points is selected. When calculating the threshold traces for a given grid point, the closest beam steering point is used. Thus, a potential beam steering loss must be taken into account when calculating the “representative” threshold for the region connected to the grid point.

▼ Architectural Design

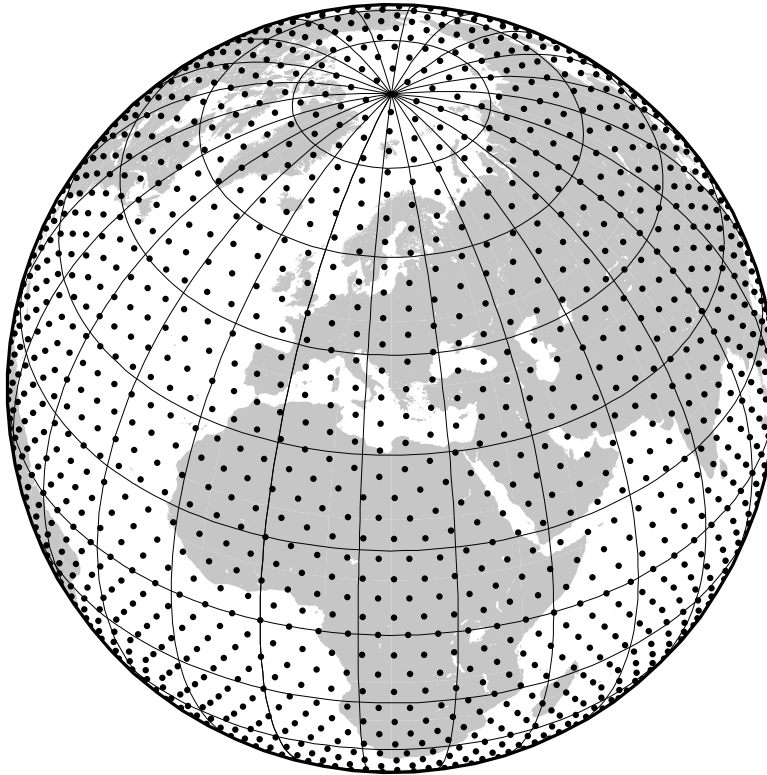


FIGURE 4. 2,562 GLOBAL MONITORING TARGET POINTS

The beam steering loss is mainly a function of array aperture and signal frequency. [Figure 5](#) shows an illustration for the ASAR array, including 3 dB beam loss contours for P-type velocities derived from analysis of ASAR events filtered in the 1–4.5 Hz filter band. These loss contours are circles when shown in slowness space, so the steering points do not translate into equidistant geographical points. If teleseismic distances mainly are considered, the number of steering points for a given worst-case loss is modest.

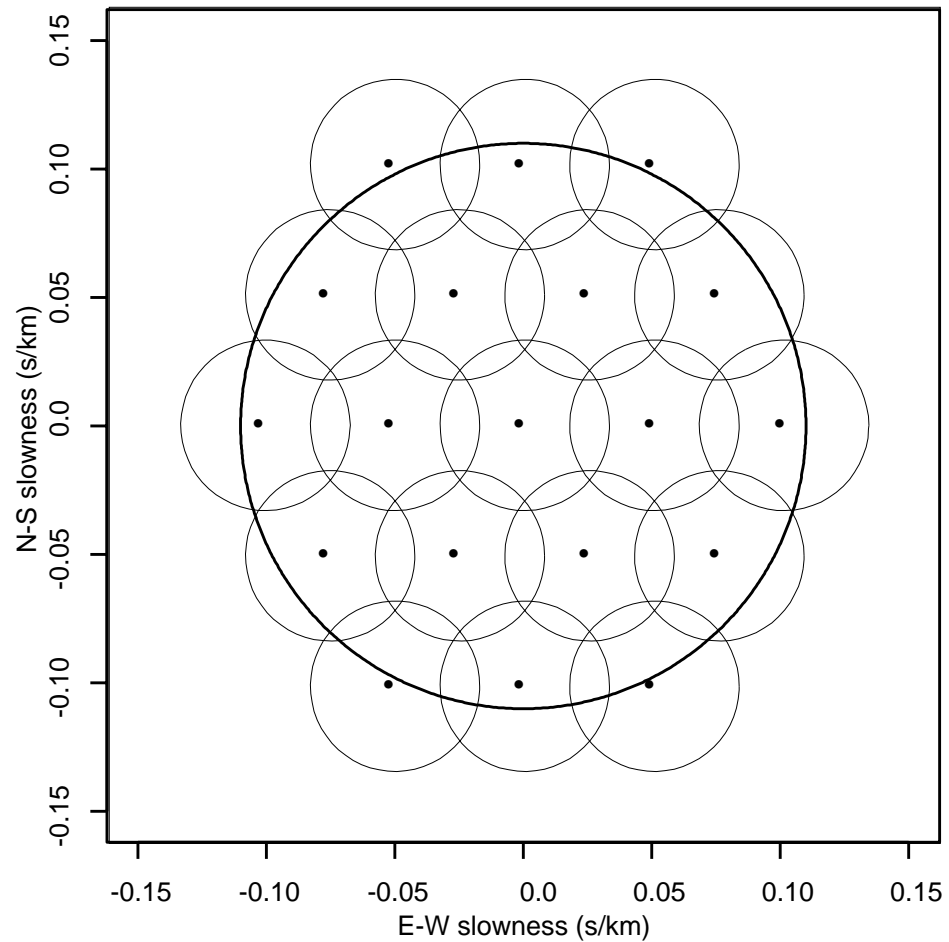


FIGURE 5. ASAR P-PHASE BEAM DEPLOYMENT

Calibrating Magnitudes and Setting Time/Azimuth/Slowness Tolerances

Optimum performance of global threshold monitoring requires access to magnitude calibration statistics for each target point and each station/phase combination considered. In a practical situation the necessary number of calibration events for each target point in the grid cannot be obtained, and a different approach is required. The approach is to consider only P-phases and to use “generic” attenua-

▼ Architectural Design

tion models. The results from several studies ([\[Vei72\]](#), [\[Rin79\]](#), and [\[Har85\]](#)) are combined to obtain the magnitude calibration function and the travel-time curves of seismic phases P, PKP_{df}, and PKP_{ab} (from IASP91 [International Association of Seismology and Physics of the Earth's Interior] [\[Ken91b\]](#), see [Figure 6](#)) over the full distance range of 0–180 degrees. For the distance range of 97–125 degrees, reliable magnitude correction parameters do not exist, so observations in this distance interval are currently not used in calculating detection capability. This lack of parameters makes little difference in practice, because the 97–125 degree distance range corresponds to the P “shadow zone” where direct P-type phases are attenuated.

In threshold monitoring, a trade-off exists between the size of the target area and the tolerances of the parameter values used in the threshold computations. With a given grid, the tolerances of each aiming point must be made compatible with the grid spacing, such that conservative estimates can be made of the detection capability.

For example, a region with a central grid point at a distance of 40 degrees from a given station will span the distance interval from 37.3 to 42.2 degrees. According to the IASP91 model [\[Ken91b\]](#), the travel-times to the closest and most distant points for a surface focus are 433.6 s and 478.6 s, respectively. Therefore, when processing this target region, the largest signal amplitude within this travel-time interval must be considered.

Similarly, variations in azimuth, slowness, and magnitude correction values within each target region must be taken into account in the processing (for more details see [\[Kvæ94\]](#)).

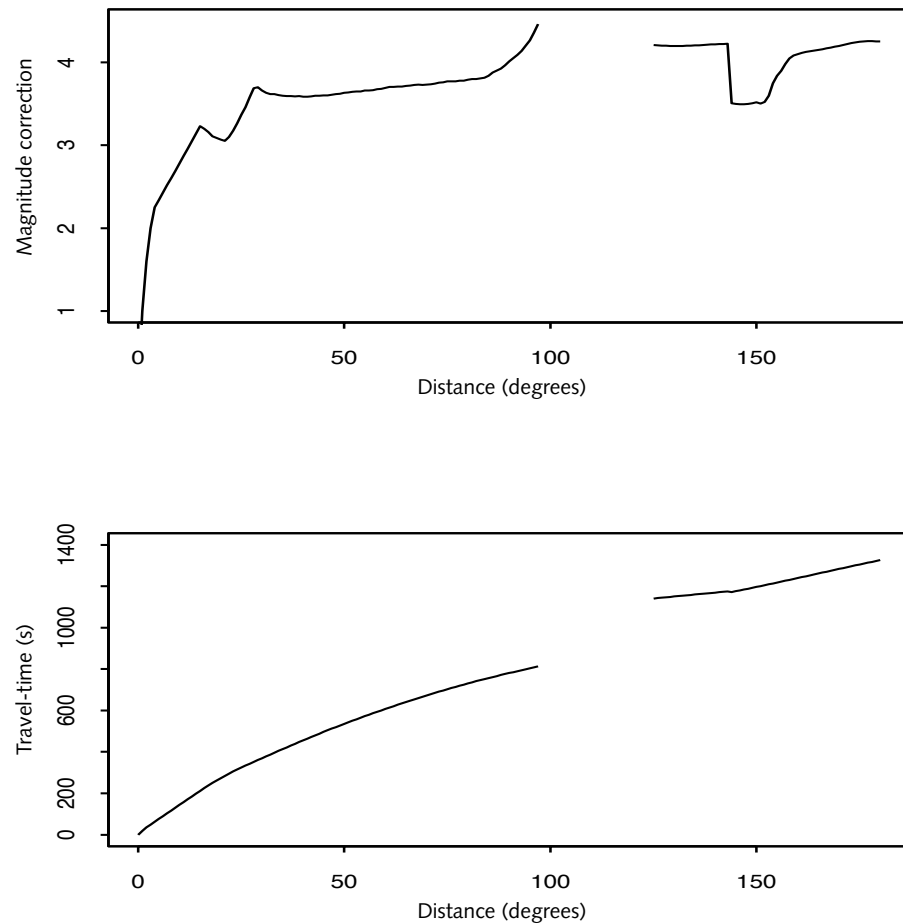


FIGURE 6. MAGNITUDE CALIBRATION FUNCTION AND TRAVEL-TIME CURVE FOR EVENTS AT ZERO DEPTH

Setting Station-specific Parameters

Traditionally, A/T is used as a representation of the signal power in estimating event magnitudes [see equation (1)]. The amplitude A is given in nanometers and the period T is given in seconds. Assuming that the instrument response is flat to velocity, the STA can be used to approximate the signal power via equation (7):

▼ Architectural Design

$$\frac{A}{T} \approx \frac{\pi}{2} \cdot \text{calib} \cdot \text{STA} \quad (7)$$

where *calib* is the system calibration constant at the reference period (usually 1 second).

In TM, the STA for arrays is measured on beams filtered with a passband suitable for estimating the event magnitude. For three-component (3-C) stations, the filtered vertical-component channel is used.

To obtain close-to-optimum processing parameters for each station, a set of 20 to 60 representative events located at different distances are typically analyzed (for details on the tuning procedure see [\[IDC6.5.14\]](#) and [\[Kvæ96\]](#)).

The following uncertainty factors are addressed in the remaining sections:

- use of STA envelopes as a representation of A/T
- effect of beamforming, filtering, and different instrument responses on the seismic amplitude
- effect of target points representing a finite geographical area and missteering of array beams

Prefilter Cutoffs

One product of TM is a map of magnitude threshold contours that show the magnitude above which events are likely to be detected. In the operations of the IDC the detection and magnitude processes use different frequency bands. The detection process is tuned to the station's characteristics (such as local noise conditions), the broadband characteristics of the seismometer, and optimal snr frequency bands. The magnitudes are calculated using a third-order Butterworth filter with a passband between 0.8 and 4.5 Hz. However, for threshold monitoring one set of prefilter cutoffs is used. These cutoffs are tuned primarily for detection. When estimating magnitude for TM, a correction is applied to the A/T estimate (see ["STA Envelopes for A/T" on page 19](#)).

Prefilter cutoffs for threshold monitoring are derived from tuning events. Low signal amplitudes during noisy conditions are used to achieve a balance between the best frequency band for high snr and the snr in the frequency band used for routine magnitude estimation. For example, ARCES is characterized by strong low-frequency noise and rather high dominant signal frequencies; a prefilter between 1.5 and 6.0 Hz is used for teleseismic monitoring above 15 degrees distance. For distances within 15 degrees a prefilter between 2.5 and 8.0 Hz is used.

STA Envelopes for A/T

For routine magnitude estimation at the IDC, a third-order Butterworth filter with a passband between 0.8 and 4.5 Hz is applied to the data prior to estimating signal amplitude and period. To resemble the procedure used for m_b calculation at the IDC, it would be ideal to apply the same filter to the data prior to generating STA envelopes. But, the prefilters used for monitoring are often different from 0.8–4.5 Hz, and some stations have system responses that are not flat to velocity in the passband. A correction to equation (7) must be introduced to use the STA when calculating the threshold parameter $a_{ij}(t)$ in equation (2). An average correction factor is calculated by comparing manually measured A/T values (0.8–4.5 Hz) of tuning event P-phases with STA values measured after prefiltering with the derived cutoffs. The correction (*corr*) is applied to the A/T estimation as shown in equation (8).

$$\frac{A}{T} \approx \frac{\pi}{2} \cdot \text{calib} \cdot \text{STA} \cdot \text{corr} \quad (8)$$

These station- and distance-specific correction factors are then used when calculating the threshold parameter for each station [see equation (2)].

Beamforming Signal Loss

For large arrays beamforming can significantly reduce the signal amplitudes, in particular for high-frequency signals. To use the measured STA signal power in a consistent way when calculating $a_{ij}(t)$ of each station, it is necessary to compensate

▼ Architectural Design

for the expected signal loss by beamforming. The expected signal loss for a station is defined as the average signal loss for the tuning events, calculated after prefiltering. The signal loss for a single event is given by equation (9):

$$\text{Signal loss} = (\text{Beam STA}) / (\text{Average STA of the individual sensors}) \quad (9)$$

where the STA used is the maximum STA in the 8 seconds after the signal onset.

As an example, regional events recorded at the ARCES array have an average signal loss of 1.5 dB after prefiltering in the frequency band 2.5–8.0 Hz.

Mis-steering Signal Loss and Beam Deployment

When deploying a beam set for processing array data, the signal loss due to mis-steering of the beams is required to be less than a given value (for example, 3 dB). If the approximate value of the slowness mis-steering (s/km) corresponding to the 3 dB signal loss is known, the steering parameters (azimuth and slowness) of the necessary beam deployment can be derived.

Information on signal loss as a function of beam mis-steering is also needed when processing a given target region (see [“Calculating Detection Capability Traces for Global Grid Systems” on page 13](#)).

Average beam loss curves of the tuning events for each array are derived by calculating the signal loss for a set of different mis-steering vectors. An example of such a curve is shown in [Figure 7](#), where events within a distance of 15 degrees of ILAR have been analyzed. In this distance range the prefilter passband 1.25–4.5 Hz is used. The average signal loss is 1.8 dB for no mis-steering, and the average 3 dB level corresponds to a mis-steering of 0.025 s/km. The light lines surrounding the dark average signal loss curve are estimates of 1σ .

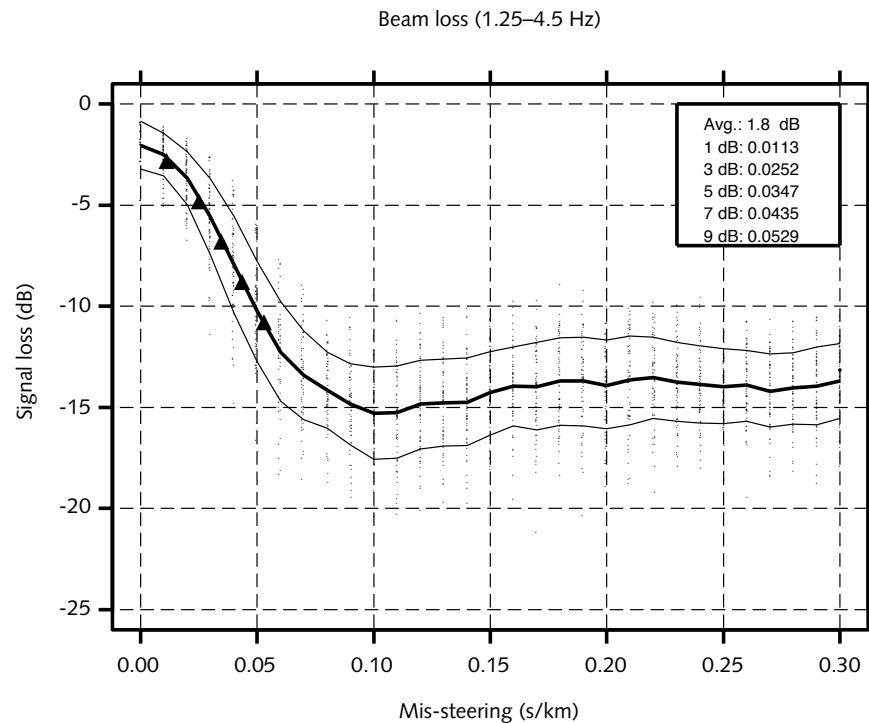


FIGURE 7. ILAR REGIONAL EVENT SIGNAL LOSS

DESIGN DECISIONS

The following design decisions pertain to TM.

Programming Language

The TM programs written in the C programming language include: *CreateTMSession*, *AddTMStation*, *DeleteTMStation*, *LoopCopy*, *rdf2cdf*, *TMthreshold*, and *TMmap*. The last three programs include FORTRAN (Formula Translation/Translator) subroutines in the program directories. *tm_stast*, *tm_beambasic*, and *tm_globrec* are entirely FORTRAN subroutines.

▼ Architectural Design

The libraries: *ctms*, *gmt*, and *netcdf* are written in C; all other libraries are written in FORTRAN, except for a few scattered C routines that are in the *tmcf*, *ngbase*, and *ngut* libraries.

All scripts (*makecdf.2562*, *TMprod*, *tmstatus*, *detplot*, *TMpostreb*, *TMbulletin*, *replotuptime*, *loopuptime*, and *CopyPSFile*) are Bourne shell scripts.

Global Libraries

The software of TM is linked to the *par* shared library. The software is also linked to the following COTS libraries: *M77*, *F77*, and *sunmath*.

Filesystem

TM uses the filesystem to store input parameters, intermediate files (such as disk loops), and results. The exact locations of these files are set as inputs to the programs (see [\[IDC6.5.14\]](#) for an example configuration).

Web

TM software does not include links to the web; however, the results of TM may be made available for viewing on the web.

Design Model

The design of TM is primarily influenced by reliability, consistency, and timeliness requirements. Reliability and consistency of the results are critical requirements.

The detection thresholds produced by TM processing must be consistent with the Reviewed Event Bulletin (REB). This consistency requirement is addressed through careful calibration of the input parameters to match the IDC bulletin processing parameters (especially the magnitude calculations).

To be reliable, TM must be both internally robust and able to interact with other IDC processes. Although the TM software rarely crashes, it does depend on other processes for its results (for example, the Detection and Feature Extraction soft-

ware [DFX]). The Distributed Application Control System (DACS), which controls most IDC processing, is also used to control TM processing. Using the DACS ensures that TM is not affected by the failure of any other process.

Although TM must deliver results in a timely manner, a limited delay is acceptable to allow for late data and completion of the REB. The DACS and *rebdone* script are used to control TM processing so that TM processing is coordinated with the availability of continuous seismic data, the Standard Event List 3 (SEL3) automated bulletin processing pipeline, and the completion of the REB.

Database Schema Overview

The TM software uses tables in the ORACLE database to extract bulletin information and to describe and announce the availability of TM products.

[Table 1](#) shows the database tables used by TM. The Name field identifies the database table. The Mode column is “R” if TM reads from the table and “W” if the system writes to the table. The Owner column indicates who has administrative control over the table. The “DBA” (database administrator) owns standard tables that are part of the core schema, and the “SS” owner indicates that the Subscription Subsystem owns the table.

TABLE 1: DATABASE TABLES USED BY TM

| Name | Mode | Description |
|----------------------|------|---|
| fileproduct | R/W | files containing individual products available for distribution |
| fpdescription | R | general description of the product types available as files |
| gregion | R | defines geographic regions |
| interval | R/W | defines units of processing |
| origerr | R | includes origin error information |
| origin | R | includes event origin information |

▼ Architectural Design

The **interval** table is used by the Detection and Feature Extraction software (DFX) when producing STA traces from the continuous seismic data. The bulletin information that is presented in the uptime product is obtained from the **gregion**, **origin**, and **origerr** tables. The **fpdescription** table must contain the descriptions of the TM file products (uptime, status, and detection) as shown in [Table 2](#). As products are produced by the TM software the individual results are entered into the **fileproduct** database table.

TABLE 2: TM FPDESCRIPTION ROWS

| typeid | prodtype | name | msgd type | msgd format | header _fpid | lddate |
|--------|--------------|----------------------|-----------|-------------|--------------|---------------|
| 1 | tm_uptime | Data Availability | bin | ps | -1 | dd- MMM-yy |
| 2 | tm_status | Station Noise Levels | bin | ps | -1 | dd- MMM-yy |
| 3 | tm_detection | Detection Capability | bin | ps | -1 | dd- MMM-yy |

Detailed descriptions of the database tables and the accounts to which they belong are provided in [\[IDC5.1.1Rev2\]](#) and [\[IDC5.1.3Rev0.1\]](#).

FUNCTIONAL DESCRIPTION

The primary functions of the TM software are to create or alter an environment for TM processing and to process the data as they arrive to obtain estimates of the network detection threshold ([Figure 8](#)). These primary functions are accomplished through a number of lower-level functions that are described in the sections that follow.

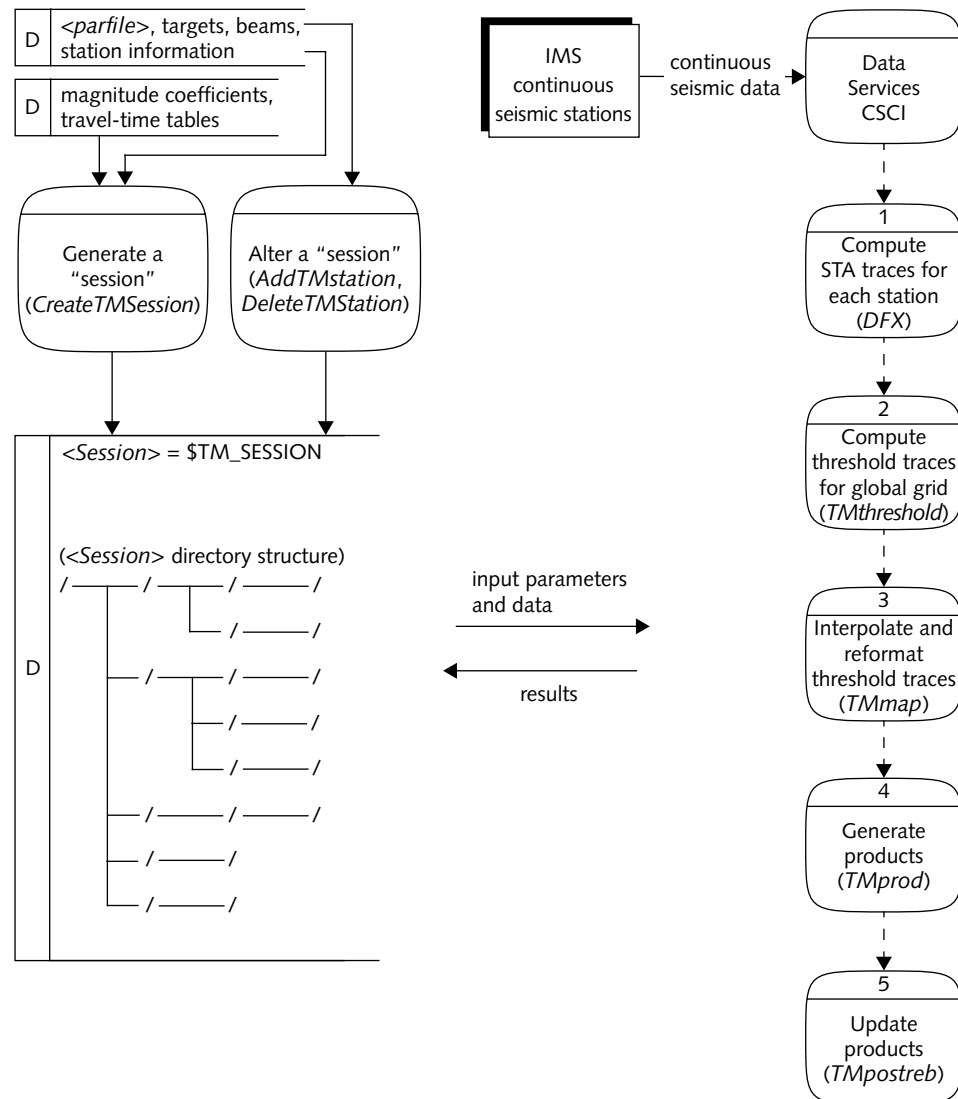


FIGURE 8. TM FUNCTIONS

Creating Processing Sessions

The directory structure and most of the files required to run TM are created by *CreateTMSession*. *DFX* must have write privileges in *<Session>* for TM to work.

▼ Architectural Design

Altering Processing Sessions

A session may be altered by adding or deleting stations using *AddTMStation* or *DeleteTMStation*, respectively. Both of these programs alter only the contents of *<Session>*. Other files must be edited manually.

Generating STA Traces for Each Station

STA traces for each station are generated by *DFX* (process 1 in [Figure 8](#)). *DFX* is part of the Station Processing CSC of the Automatic Processing CSCI. Continuous seismic data from the Continuous Data Subsystem ([\[IDC6.5.18\]](#) and [\[IDC7.4.1\]](#)) are processed to generate STA traces in *\$TM_SESSION/Region/STA/*. [\[IDC6.5.14\]](#) describes how *DFX* is configured for TM.

Generating Threshold Traces for Global Grid

TMthreshold (process 2 in [Figure 8](#)) reads the STA traces generated for TM by *DFX*. These are processed according to the recipes found in *\$TM_SESSION/Region/TM/TM_recipes/*. The files are written to disk loops (one for each of 2,562 targets) in *\$TM_SESSION/Region/TM/<tmttype>/Data/*.

Interpolating and Reformatting Threshold Traces for Global Grid

TMmap (process 3 in [Figure 8](#)) reads the target disk loops (*\$TM_SESSION/Region/TM/<tmttype>/Data/<target>*) and generates an American Standard Code for Information Interchange (ASCII) file that can be read by GMT. It executes *mak-ecdf.2562*, which then runs the GMT routine *surface*; the results of this are then written back to a disk loop (*\$TM_SESSION/Region/TM/<tmttype>/RDF/grid.rdf*).

Generating Products

The Bourne shell script *TMprod* (process 4 in [Figure 8](#)) calls three scripts to generate products. The script *tmstatus* generates status plots using *tm_astat* to read the STA data and GMT routines to generate the plots. The script *plotuptime* generates a map showing the stations and their percent of uptime using GMT routines to make the maps. The script *detplot* reads the Regional Data Format (RDF) disk loop (\$TM_SESSION/Region/TM/Network/RDF/grid.rdf) with *rdf2cdf* and generates the global threshold maps with GMT routines. Information is inserted into the **fileproduct** database table by all three scripts: *tmstatus*, *plotuptime*, and *detplot*.

Updating Products with Event Information

TMpostreb (process 5 in [Figure 8](#)) reads REB information (with script *TMbulletin*) from the **region**, **origin**, and **origerr** database tables and writes an ASCII file in \$TM_SESSION/Bulletin/TMbulletin.IDCREB. *plotuptime* is then called to read the ASCII file, regenerate the uptime plot adding event origin information, and insert product information into the **fileproduct** database table.

[Table 3](#) lists the products and availability times for the TM software. All TM products are available about 11 hours behind real time, but the uptime plot will not have event information included until the REB has been completed. [Table 4](#) defines terms relating to standard products.

TABLE 3: STANDARD PRODUCTS AVAILABLE THROUGH TM

| Product | Availability |
|---------|---|
| status | hourly, 11 hours behind real time |
| uptime | hourly, 11 hours behind real time (event information is added after the REB is completed) |
| detplot | hourly, 11 hours behind real time |

▼ Architectural Design

TABLE 4: TERMS RELATING TO STANDARD PRODUCTS

| Name | Description |
|------------|---|
| bulletin | A chronological listing of event origins spanning an interval of time. Often, the specification of each origin, or event, is accompanied by the event's arrivals, and sometimes with the event's waveforms. |
| detplot | A TM product that shows a map of the average and worst case probability of detection for the primary seismic network of the IMS over the one-hour interval covered by the plot. |
| event list | A chronological listing of event origins spanning an interval of time. Unlike a bulletin, an event list is not reviewed by analysts; it is produced automatically. |
| status | A TM product that includes a plot of the short-term average traces for each primary seismic station for the one-hour interval covered by the plot. |
| uptime | A TM product that shows a global map of the primary seismic stations, color-coded by the percentage of time for which data are available from each station, for the one-hour interval covered by the plot. REB events that occurred in the one-hour interval are added to the plot after the REB is produced. |

INTERFACE DESIGN

This section describes TM's interfaces with other IDC systems, external users, and operators.

Interface with Other IDC Systems

TM relies on the Continuous Data Subsystem in the Data Services CSCI to provide data from the primary seismic stations of the IMS. These data are processed to form STA traces using the Station Processing CSC of the Automatic Processing CSCI. The Distributed Processing CSCI is used to control TM processing as part of the SEL3 pipeline. Because *TMpostreb* depends on the REB, the Interactive Processing CSCI influences TM processing. The products of TM processing are made available to the Data Services CSCI for distribution through database entries in the **fpdescription** database table.

Interface with External Users

The products of TM are available to external users through the Data Services CSCI. The specific mechanisms might include: *AutoDRM*, subscription, or the website.

Interface with Operators

Normally, TM is invoked by the station processing pipeline, the SEL3 pipeline, and the *rebdone* script. Pipelines are controlled by the DACS, and the *rebdone* script is initiated manually when the bulletin review has been completed. The performance of the TM software is monitored through log files. Error messages, their meaning, and the appropriate actions to take are included in the user manual [\[IDC6.5.14\]](#).

Chapter 3: Detailed Design

This chapter describes the detailed design of TM and includes the following topics:

- [Data Flow Model](#)
- [Processing Units](#)
- [Database Description](#)

Chapter 3: Detailed Design

DATA FLOW MODEL

Data for TM calculations are the continuous seismic data from the stations of the primary seismic network, which are recorded on disk loops (D in [Figure 9](#)) by the Data Services CSCI. The seismic data for each station are checked for quality, beamformed (arrays only), bandpass filtered, and short-term averaged using *DFX* ([\[Wah96\]](#)). The *DFX* program is run in the station processing pipeline, processing 10 minute data segments as soon as complete segments are recorded and available at the IDC. The continuous STA data are then stored on new disk loops with a typical sampling interval of one second. Data gaps and processing gaps are identified by NULL values in the disk loops.

Network detection thresholds are calculated by *TMthreshold* (process 1 in [Figure 9](#)). This program reads the STA disk loops for each of the primary seismic stations, calculates the 90 percent detection thresholds for each of 2,562 globally distributed target areas at 10-second intervals, and stores the results on disk loops. *TMthreshold* is run in the SEL3 pipeline, which initiates processing of 20-minute segments with a time delay of 10 hours behind real time.

To facilitate map visualization of the results, the threshold traces for each time sample are interpolated, reformatted, and written to a new disk loop using a program called *TMmap* (process 2 in [Figure 9](#)). *TMmap* is run in the SEL3 pipeline after *TMthreshold* has completed the detection threshold calculations for the given time segment.

STA trace and network detection threshold calculations are followed by the calculation of three sets of hourly results from TM. These results are calculated by a Bourne shell script called *TMprod* (process 3 in [Figure 9](#)), which is run in the SEL3 pipeline.

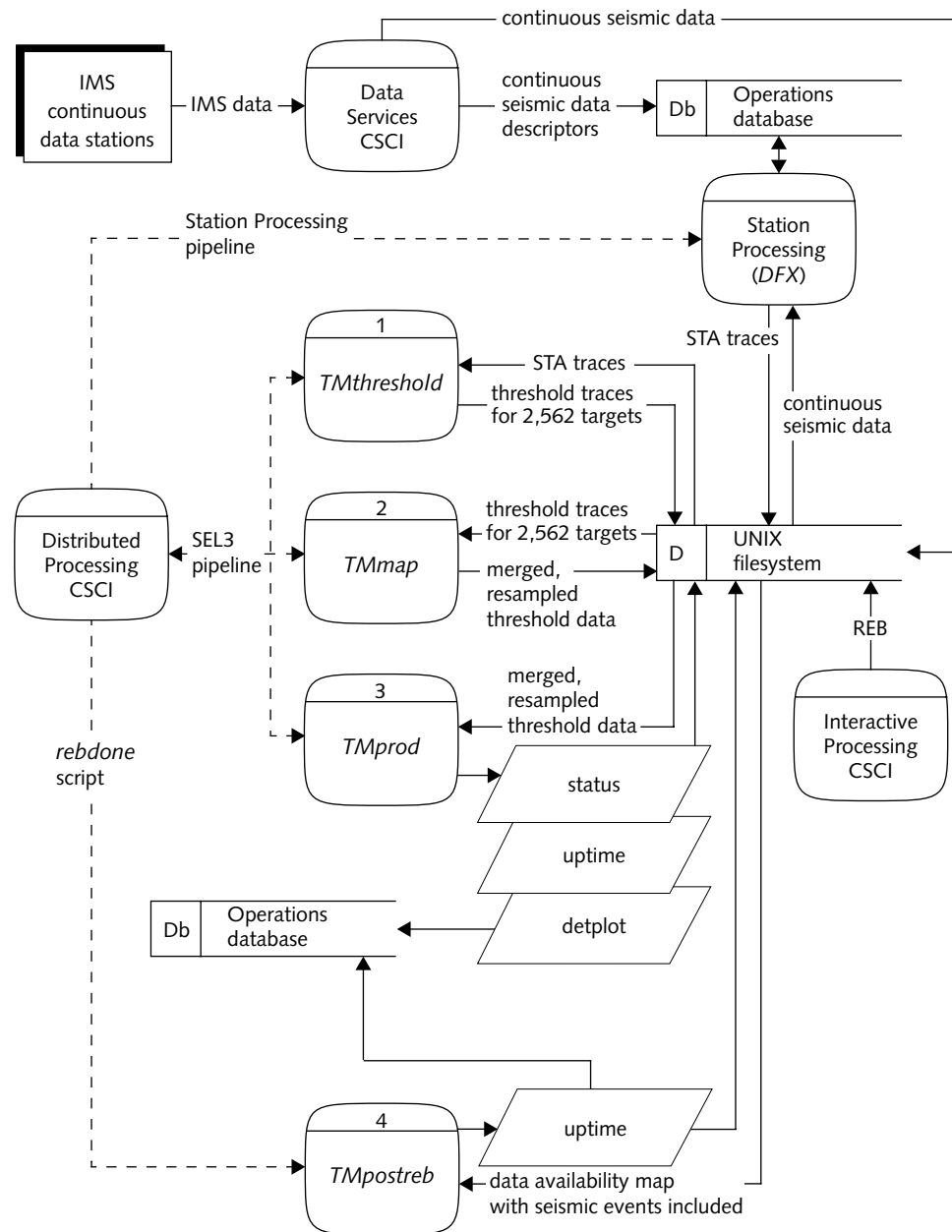


FIGURE 9. TM DATA FLOW MODEL

▼ Detailed Design

When the REB is complete for a given day, data availability maps are recreated to include the REB event information. This is done with a Bourne shell script called *TMpostreb* (process 4 in [Figure 9](#)), which is invoked in the *rebdone* script.

PROCESSING UNITS

TM consists of the following processing units:

- *CreateTMSession*
- *AddTMStation*
- *DeleteTMStation*
- *TMthreshold*
- *TMmap*
- *TMprod*
- *TMpostreb*
- *LoopCopy*

The following paragraphs describe the design of these units, including any constraints or unusual features in the design. The logic of the software and any applicable procedural commands are also provided.

CreateTMSession

The directory structure and most of the files required to run TM are created by *CreateTMSession*.

Input

CreateTMSession has a number of arguments. Although it is possible to enter every argument on the command line, it is most convenient to use a parameter file. The par file may contain the parameters shown in [Table 5](#). Parameters in parameter files may be overridden by placing them after the `par=<parfile>` specification on

the command line. The same format is used for the par file and the command line. The search order is from left to right on the command line, with precedence given to parameters found later if there are multiple specifications.

TABLE 5: CREATETMSESSION PARAMETERS

| Parameter | Default | Description |
|--------------------------|-----------------|---|
| <i>session_directory</i> | none (required) | fully qualified path of session directory <i>Session</i> to be created (normally \$TM_SESSION for pipeline processing) |
| <i>tmttype</i> | Network | currently, <i>tmttype</i> should always be Network |
| <i>phases</i> | PTM | comma-separated list of phases to process (for example, <i>phase1, phase2,..., phasen</i>) |
| <i>stlist</i> | none (required) | comma-separated list of stations to use (for example, <i>stn1,stn2,..., stnn</i>) |
| <i>staticdir</i> | \$TM_HOME/data | fully qualified path of static data directory <i>staticdir</i> |
| <i>ntargets</i> | 2562 | number of targets to use (considered a string by the program and used to determine which targets file to copy) |
| <i>staloop</i> | none (required) | duration (s) of the STA disk loops (for example, TM001) |
| <i>stasamp</i> | 1.0 | time (s) between the centers of successive STA windows |
| <i>stalen</i> | 1.0 | length of time (s) of the STA window |
| <i>stanullval</i> | -2.0 | value to be used in STA data files where there are no data |
| <i>tmloop</i> | none (required) | length of time (s) of the TM disk loops (for example, Target_1001) |
| <i>tmsamp</i> | 10.0 | length of time (s) between the centers of successive TM windows |
| <i>tmlen</i> | 1.0 | length of time (s) of the TM window |
| <i>tmnullval</i> | -2.0 | value to be used in TM data files where there are no data |

▼ Detailed Design

TABLE 5: CREATETMSESSION PARAMETERS (CONTINUED)

| Parameter | Default | Description |
|-----------------|---------|--|
| <i>nxgrid</i> | 91 | size (pixels) of the <code>grid.rdf</code> file (x axis) |
| <i>nygrid</i> | 46 | size (pixels) of the <code>grid.rdf</code> file (y axis) |
| <i>magstdev</i> | 0.38 | standard deviation assumed for the P magnitude estimates |
| <i>magconf</i> | 0.90 | confidence level of upper magnitudes limits |
| <i>search1</i> | 6.0 | length of time (s) of search interval after predicted P wave onset |
| <i>search2</i> | 3.0 | additional search interval (s) due to travel-time uncertainties |

The *staticdir* parameter is used to access several files that are used by *CreateTMSession*. These files include:

```
staticdir/Targets/targets.2562
staticdir/Beamset/*
staticdir/libdata/TM.site
staticdir/libdata/magtab1/tab.PTM
staticdir/libdata/trtab1/tab.PTM
```

The file *staticdir/Targets/targets.n*, where *n* is the number of targets, is an ASCII file containing a table of the parameters listed in [Table 6](#).

TABLE 6: TARGETS.N PARAMETERS

| Column and Parameter | Description |
|----------------------|---|
| 1 <i>name</i> | target name (Target_seqnum, for example, Target_1001) |
| 2 <i>lat</i> | latitude (deg., for example, 26.72) |
| 3 <i>lon</i> | longitude (deg., for example, -162.00) |

TABLE 6: TARGETS.N PARAMETERS (CONTINUED)

| Column and Parameter | Description |
|----------------------|---|
| 4 <i>depth</i> | depth (km, for example, 0.00) |
| 5 <i>radius</i> | radius of the target region (deg., for example, 2.74) |
| 6 <i>seqnum</i> | sequence number (beginning with 1001) |

A subdirectory for each station in the network is contained in *staticdir/Beamset/*. Each subdirectory contains a file with beam information (*beambasic*) and one or more files with steering parameters relevant to the station. The *beambasic* file has one heading row and one row for each beam. The parameters in *beambasic* (presented as columns of a table with a header) are described in [Table 7](#).

TABLE 7: BEAMBASIC PARAMETERS

| Column and Parameter | Description |
|----------------------|---|
| 1 <i>Name</i> | name of current beam (for example, TM001) |
| 2 <i>Type</i> | either "B" (beam) or "S" (single channel) |
| 3 <i>Azi</i> | azimuth for this beam (deg.) |
| 4 <i>Slow</i> | slowness for this beam (s/km) |
| 5 <i>STAstp</i> | STA sampling rate (sample/s) |
| 6 <i>STAlen</i> | length of STA window (s) |
| 7 <i>Dlow</i> | closest distance from station for this beam (deg.) |
| 8 <i>Dhig</i> | farthest distance from station for this beam (deg.) |
| 9 <i>DFXbeam</i> | actual beam (written by <i>DFX</i>), which may differ from <i>Name</i> |
| 10 <i>Steerpar</i> | file containing steering parameters for this beam |
| 11 <i>Ref</i> | name of reference station (usually the current station) |
| 12 <i>Freq1</i> | low end of filter bandpass (Hz) |
| 13 <i>Freq2</i> | high end of filter bandpass (Hz) |

▼ Detailed Design

Files listed in column 10 of `beambasic` have names with the form:

`missteer.Dlow-Dhig`

where *Dlow* and *Dhig* are found in columns seven and eight of `beambasic`. [Table 8](#) describes the contents of the `missteer.Dlow-Dhig` files. Single channel beams cannot include mis-steering or signal loss.

TABLE 8: MISSTEER.DLOW-DHIG PARAMETERS

| Parameter | Description |
|----------------------|--|
| <i>A/Tcomp</i> | average difference between $\log(A/T)$ between 0.8 and 4.5 Hz and $\log((\pi/2) \cdot STA \cdot calib)$ in three different frequency bands for events more than 15 deg. from the station |
| <i>sloss</i> | average reduction of signal amplitude after beamforming due to signal decorrelation (dB) |
| detailed signal loss | column 1: beam mis-steering (s/km) column 2: signal loss (dB) |

The `staticdir/libdata/` directory includes `TM.site`, which is a dump of the IDC site database table. `TM.site` contains coordinate information for the network and is a read-only ASCII file with one row per site ([Table 9](#)).

TABLE 9: TM.SITE PARAMETERS

| Column and Parameter | Description |
|----------------------|---------------------|
| 1 <i>sta</i> | station identifier |
| 2 <i>ondate</i> | Julian start date |
| 3 <i>offdate</i> | Julian off date |
| 4 <i>lat</i> | latitude (deg.) |
| 5 <i>lon</i> | longitude (deg.) |
| 6 <i>elev</i> | elevation (km) |
| 7 <i>staname</i> | station description |

TABLE 9: TM.SITE PARAMETERS (CONTINUED)

| Column and Parameter | Description |
|----------------------|---|
| 8 <i>statype</i> | station type: <i>ss</i> = single station, <i>ar</i> = array |
| 9 <i>refsta</i> | reference station for array members |
| 10 <i>dnorth</i> | offset from array reference (km) |
| 11 <i>deast</i> | offset from array reference (km) |
| 12 <i>lddate</i> | load date |

Attenuation curves are tabulated in the ASCII file *tab.PTM*, which is located in the *magtab1* subdirectory of *staticdir/libdata/*. An ASCII travel-time table, also called *tab.PTM*, is found in the *trtab1* subdirectory of *staticdir/libdata/*. *PTM* is the only phase for which attenuation curves and travel-time tables are included in *staticdir/*. [Table 10](#) describes the parameters of the *tab.PTM* files.

TABLE 10: TAB.PTM PARAMETERS

| Parameter | Description |
|--|---|
| <i>filename</i> | table to use for teleseismic P waves (<i>n</i> for N/A) |
| <i>ndepth</i> | number of depth samples |
| <i>depths</i> | blank separated depths (km) |
| <i>ndist</i> | number of distance sample |
| <i>distances</i> | blank separated distances (deg.) |
| <i>data(depth[1],distance[i])</i> | Q or travel-time/amplitude for depth 1, one row per distance |
| ... | ... |
| <i>data(depth[ndepth],distance[i])</i> | Q or travel-time/amplitude for depth <i>ndepth</i> , one row per distance |

▼ Detailed Design

Output

CreateTMSession generates a directory structure for threshold monitoring processing. [Figure 10](#) shows the directory structure, *Session*. The following paragraphs describe the structure.

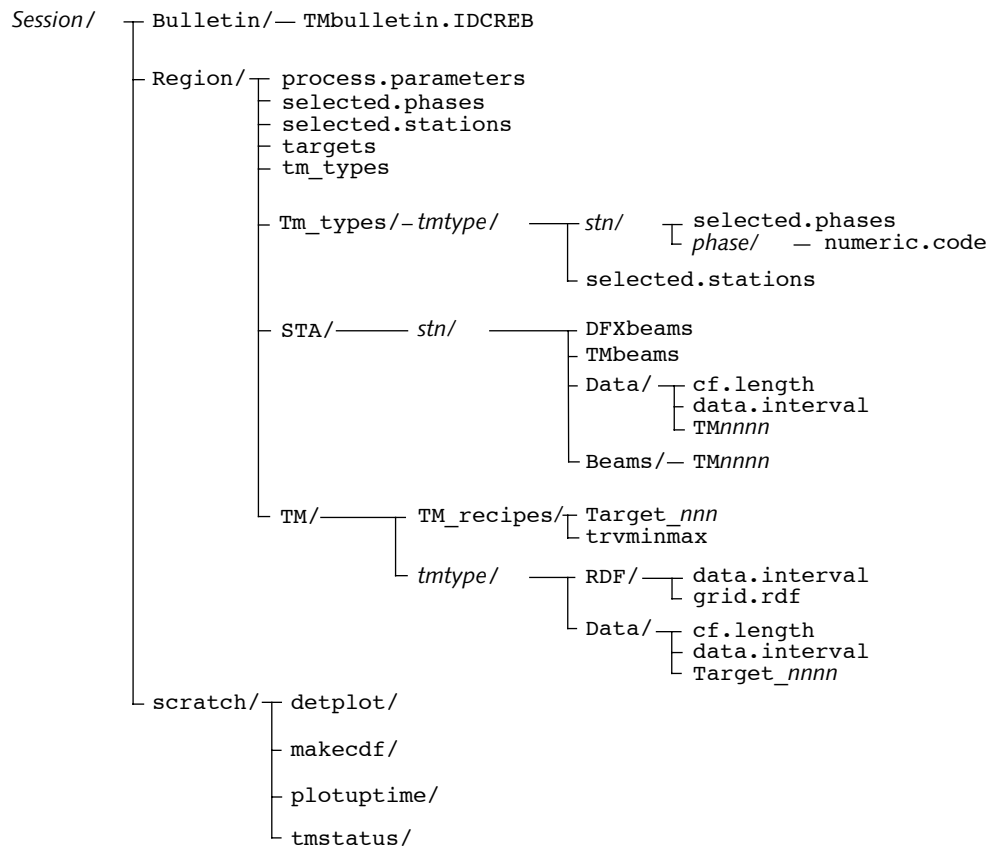


FIGURE 10. SESSION DIRECTORY STRUCTURE

The reviewed IDC bulletin for the two weeks preceding the run time is located in `Bulletin/` (set in *TMpostreb*). *TMbulletin* creates the bulletin in a temporary file and then moves the temporary file to `Bulletin/` overwriting the old file. The name of this file is `TMbulletin.IDCREB` (also set in *TMpostreb*). Seismic events in this bulletin are plotted on the station availability map.

The scripts *makecdf.2562*, *tmstatus*, *plotuptime*, and *detplot* write temporary files in the corresponding subdirectories in `scratch/`.

Five ASCII files in `Region/` contain information pertinent to the processing session. The first file, `process.parameters`, contains the parameters *magstdev*, *magconf*, *search1*, *search2*, *tmsamp*, *tmlen*, *stasamp*, and *stalen* written in free format (see [Table 5 on page 35](#) for a description of these parameters). The file `targets` is a copy of the appropriate target list described in [Table 6 on page 36](#).

The remaining three files are in a two-column format with a name in the first column and an index number (line number) in the second column:

| | |
|--------------------------------|--|
| <code>selected.stations</code> | lists the stations of interest (for example, ABKT, ARCES, and so on) |
| <code>selected.phases</code> | lists the phases of interest (for example, PTM) |
| <code>tm_types</code> | lists the “tmtypes” of interest (currently only Network) |

The `Region/STA/` directory includes a *stn* subdirectory for each station of interest (for example, ARCES), each having two subdirectories (`Beams/` and `Data/`). Each *stn/* directory also includes two ASCII files: `TMbeams` and `DFXbeams`. `DFXbeams` contains the list of STA beams written by the *DFX* program. `TMbeams` contains the list of TM beam definitions, which (for arrays) includes mis-steering parameters.

Names of beams (and corresponding files in *stn/Beams/* and *stn/Data/*) follow the format:

TMnnn

▼ Detailed Design

where *nnn* is a three digit sequence number, with leading zeroes if needed (for example, 001).

The STA traces for each station are written in disk loops found in *stn/Data/*. The lengths of these files (in seconds) are found in *stn/Data/cf.length*. The starting and ending times (in format *YYYY-DDD:HH.MM.SS.sss*) are found in *stn/Data/data.interval*.

The ASCII processing recipes included in the files listed in *TMbeams* are found in *stn/Beams/* and contain the parameters listed in [Table 11](#).

TABLE 11: BEAMS/TMNNN PARAMETERS

| Line | Column | Description |
|------|--------|--|
| 1 | 1 | beam type (B for beam, S for single) |
| | 2 | azimuth of beam (deg.) |
| | 3 | slowness of beam (s/km) |
| 2 | 1 | STA file from <i>DFX</i> to be used for this beam (for example, <i>TM001</i>) |
| | 2 | reference channel for beam from site table (for example, <i>ARCES</i>) |
| 3 | 1 | step size between centers of successive STA windows (s) |
| | 2 | duration of STA window (s) |
| 4 | 1 | lower limit for distances covered by this beam (deg.) |
| | 2 | higher limit for distances covered by this beam (deg.) |
| 5 | 1 | magnitude bias due to filtering and STA calculation (m_b units) |
| | 2 | assumed signal loss (dB) at correct steering |
| 6–N | 1 | beam mis-steering slowness (s/km) for signal loss |
| | 2 | signal loss (dB) at the beam mis-steering slowness given in column 1 |

Region/Tm_types/ includes subdirectories for each “*tmttype*” (currently, *Network* is the only *tmttype*), each having a full complement of *stn/* subdirectories. Each of the *stn/* subdirectories has one *phase/* subdirectory for each phase of interest (for example, *PTM*). *tmttype/* contains a copy of the *selected.stations*

file, and each *tmttype/stn/* directory contains a copy of *selected.phases*. Every *tmttype/stn/phase/* directory contains a *numeric.code* file that contains a code calculated using the index numbers of the current station and phase (from *selected.stations* and *selected.phases*) as follows:

$$\text{code} = (\text{stn_index}) \cdot 10 + \text{phase_index}$$

This value is always a three-digit number with a leading zero if necessary.

Region/TM/TM_recipes/ contains an ASCII file for each target named *Target_nnnn*, where the sequence numbers (*nnnn*) begin with 1001. [Table 12](#) shows the format of the *Target_nnnn* files.

TABLE 12: TM_RECIPES/TARGET_NNNN PARAMETERS

| Line | Column | Description |
|----------------|--------|--|
| 1 | 1 | sequence number (for example, 1001) |
| | 2 | target name (for example, Target_1001) |
| | 3 | latitude (deg.) of the target |
| | 4 | longitude (deg.) of the target |
| | 5 | depth (km, usually 0) of the target |
| | 6 | radius of the target region (deg.) |
| 2 ¹ | 1 | numeric code (from the <i>numeric.code</i> file, for example, 011 for station 1 and phase 1) |
| | 2 | fully specified directory of STA trace (for example, <Session>/Region/STA/stn/Data) |
| | 3 | name of the data file for the beam (for example, TM001) |
| | 4 | station (for example, NORES) |
| | 5 | phase (for example, PTM) |
| | 6 | distance from region to center of station (deg.) |

▼ Detailed Design

TABLE 12: TM_RECIPES/TARGET_NNNN PARAMETERS (CONTINUED)

| Line | Column | Description |
|---------------------------|--------|--|
| 3 ¹ | 1 | travel-time (s) between station and center of target region for <i>phase</i> |
| | 2 | travel-time tolerance (s), lower limit |
| | 3 | travel-time tolerance (s), higher limit |
| | 4 | <i>TMlen</i> (length of TM window, s) |
| | 5 | amplitude correction factor (added to log A/T to estimate magnitude) |
| | 6 | assumed standard deviation of magnitude estimate |
| 2N+2 | 1 | index number |
| | 2 | tmtype (from Region/tm_types) |
| 2N+3 through 2N+3+M | 1 | numeric codes |
| 2N+M+4 | 1 | output directory |
| | 2 | output file |

1. Lines 2 and 3 are repeated for each of the N STA traces to be used for the target. Station lines may be repeated if a target references several beams; the largest of the amplitudes from the beams is used for the target.

Along with the 2,562 `Target_nnnn` files, `trvminmax` is an ASCII table of travel-time intervals for each station. [Table 13](#) shows the columns of the file. The travel times are adjusted for the radius of the target region (this adjustment can result in a negative number of seconds).

TABLE 13: TM_RECIPES/TRVMINMAX PARAMETERS

| Column | Description |
|--------|---|
| 1 | station name |
| 2 | station index number (as it would appear in the file <code>selected.stations</code>) |
| 3 | minimum travel time (s) |
| 4 | maximum travel time (s) |

`Region/TM/tmtype/RDF/` contains `grid.rdf`, a binary file that is a disk loop for storing the TM maps. The start and end times (`YYYY-DDD:HH.MM.SS.sss`) relevant to this file are listed in `data.interval` (in the same directory).

Detection thresholds for each target are stored in the disk loops named `Target_nnnn`. The lengths of these files are stored in `cf.length`, and the relevant start and end times are found in `data.interval` (with format `YYYY-DDD:HH.MM.SS.sss`). These two files are found in `tmtype/Data/`, along with the `Target_nnnn` files.

Control

`CreateTMSession` is run by the operator from a control line. Execution terminates if an error is detected or when the session has been created.

Error States

`CreateTMSession` has optional “verbose” modes that cause informative messages to be printed out as the program runs. For example, using the `verbose` option will display the name of each file and directory created, except for the disk loops. Also, there are a few messages from some of the FORTRAN programs and subroutines, which are called by `CreateTMSession`. Using `verbose=2` makes it possible to print

▼ Detailed Design

more information, including the name of each disk loop file created (there may be thousands). Additional information from the FORTRAN programs (which is primarily useful for debugging) may be printed out with higher levels of verbosity.

Specific error messages from *CreateTMSession* are listed in [\[IDC6.5.14\]](#).

AddTMStation

AddTMStation adds new stations to the TM processing environment.

Input/Processing/Output

AddTMStation uses a subset of the parameters used for *CreateTMSession*. Although it is possible to enter every parameter on the command line, it is most convenient to use the parameter file for *CreateTMSession*. The par file may contain all of the parameters shown in [Table 5 on page 35](#). *AddTMStation* requires the *session_directory*, *stlist*, and *staloop* parameters. The *stlist* parameter gives the list of stations to be added to the processing session. If the *stlist* parameter is entered on the command line after *par=<parfile>*, it need not be changed in the parameter file. *AddTMStation* also uses *staticdir*, *stasamp*, and *stalen* but the defaults shown in [Table 5](#) are assumed if they are not given.

AddTMStation compares the list of new stations to the list of existing stations found in *selected.stations*, and will only add those that are not duplicates.

The output of the *AddTMStation* process is an altered version of the directory structure, *Session*, shown in [Figure 10 on page 40](#) and described in the text following the figure.

Control

AddTMStation is run by the operator from the control line. Execution terminates if an error is detected or when the *Session* directory structure has been altered.

Error States

AddTMStation has the same (optional) verbosity parameter as *CreateTMSession*. Specific error messages from *AddTMStation* are listed in [\[IDC6.5.14\]](#).

DeleteTMStation

DeleteTMStation removes stations from the TM processing environment.

Input/Processing/Output

DeleteTMStation uses the same parameters used for *CreateTMSession*. Although it is possible to enter every parameter on the command line, it is most convenient to use the parameter file for *CreateTMSession*. The par file may contain all of the parameters shown in [Table 5 on page 35](#). The *stlist* parameter gives the list of stations to be removed from the processing session. If the *stlist* parameter is entered on the command line after *par=<parfile>*, it need not be changed in the parameter file.

DeleteTMStation compares the new station list specified by the *stlist* parameter to the old station list in *selected.stations* and only attempts to remove those stations that appear in both lists. The entire *<Session>/Region/Tm_types/* directory structure and *selected.stations* shown in [Figure 10 on page 40](#) are rewritten, and the TM processing recipes in *<Session>/Region/<tmttype>/TM_recipes* are recalculated.

Control

DeleteTMStation is run by the operator from the control line. Execution terminates if an error is detected or when the *Session* directory structure has been altered.

Error States

DeleteTMStation has the same (optional) verbosity parameter as *CreateTMSession*. Specific error messages from *DeleteTMStation* are listed in [\[IDC6.5.14\]](#).

TMthreshold

TMthreshold calculates the 90 percent detection thresholds for each of 2,562 globally distributed target areas. A system-wide C front end was added to this program so it could use the *getpar/mstpar* routines; the remainder of the software is written in FORTRAN except for a very few low-level C routines in the *tmcf* library.

Input/Processing/Output

Although it is possible to enter every argument on the command line, it is most convenient to use a parameter file. The par file may contain the parameters shown in [Table 14](#). Parameters in parameter files may be overridden by placing them after the `par=<parfile>` specification on the command line. The same format is used for the par file and the command line. The search order is from left to right on the command line, with precedence given to parameters found later if there are multiple specifications.

TABLE 14: TMTHRESHOLD PARAMETERS

| Parameter | Default | Description |
|--------------------------|--------------------|---|
| <i>session_directory</i> | none (required) | location of the TM processing environment |
| <i>t1</i> | none (required) | start time of the processing segment |
| <i>t2</i> | none (required) | end time of the processing segment |
| <i>verbose</i> | none (optional) | if present, diagnostics will be printed on stdout |

TABLE 14: TMTHRESHOLD PARAMETERS (CONTINUED)

| Parameter | Default | Description |
|----------------|--------------------|---|
| <i>method</i> | none (required) | processing method, either <i>detection</i> (calculate 90 percent detection threshold) or <i>upplim</i> (calculate 90 percent upper magnitude limit for non-detected events) |
| <i>ndetsta</i> | 3 | number of stations required for event detection (used only if <i>method=detection</i>) |
| <i>detsnr</i> | 4.0 | snr required for phase detection for all stations (used only if <i>method=detection</i>) |

The detailed processing flow for *TMthreshold* and its main subroutine *regproc* is described in the following paragraphs:

1. Subroutine *rdtarsstm* is called to read the following files:
 - `$TM_SESSION/Region/selected.stations` (list of stations)
 - `$TM_SESSION/Region/selected.phases` (list of phases)
 - `$TM_SESSION/Region/tm_types` (list of TM types)
2. The lists of stations and phases contained in these files are compared to those in the following directories:
 - `$TM_SESSION/Region/Tm_types/tmtype/selected.stations`
 - `$TM_SESSION/Region/Tm_types/tmtype/stn/
selected.phases`
3. `$TM_SESSION/Region/Tm_types/tmtype/stn/phase/numeric.code` is read.
4. The target coordinates are read into a common block from `$TM_SESSION/Region/targets`.
5. *regproc* calls *gmtwrite* to write the current time to `$TM_SESSION/Region/TM/tmtype/Data/GMT.last` indicating that *TMthreshold* is running.
6. *regproc* calls *rdprrec* to read the processing parameters from `$TM_SESSION/Region/process.parameters`.

▼ Detailed Design

7. *regproc* calls *rdtmrec* to read both the target recipe information from `$TM_SESSION/Region/TM/TM_recipes/compact` (if it exists) and the minimum and maximum travel-times from `$TM_SESSION/Region/TM/TM_recipes/trvminmax`. If no compact file is present, then the target recipes are read from `$TM_SESSION/Region/TM/TM_recipes/Target_nnnn` and a new compact file is written out.
8. The recipe information (travel-times, and so on) is stored in a common block.
9. *regproc* calls *rddataint* to read the start and stop times for the STA data currently found on the disk loops from `$TM_SESSION/Region/STA/stn/Data/data.interval`.
10. The lengths of the disk loops are read from `$TM_SESSION/Region/STA/stn/Data/cf.length`.
11. If any stations are to be masked, a file containing the station names, one per line, is read from `$TM_SESSION/Region/TM/TM_recipes/stations.mask` (if not found, no stations are masked).
12. The start and stop times for the target disk loops are read from `$TM_SESSION/Region/TM/tmtype/Data/data.interval`, and the lengths of these disk loops are read from `$TM_SESSION/Region/TM/tmtype/Data/cf.length`.
13. Extensive checks are performed on the time intervals, and a decision is made as to whether to process. If processing continues, *gmtwrite* is called again (as will be done repeatedly) to update the current time in `$TM_SESSION/Region/TM/tmtype/Data/GMT.last`.
14. *regproc* calls *rprocess*, which divides the data interval into chunks, and reads the STA data from `$TM_SESSION/Region/STA/stn/Data/TMnnn` with *tmstacread* (in the *tmcf* library).
15. For each data chunk that is processed:
 - Average amplitudes are found with *getavgmean* (*method* = *detection*) or *getavgmax* (*other method*).
 - Upper magnitude limits are calculated with *magclc*.

- *tmcfwrite* (in the *tmcf* library) writes data to the target disk loops `$TM_SESSION/Region/TM/tmtype/Data/Target_nnnn`.
 - *dintwrite* writes the new time interval for these target disk loops to `$TM_SESSION/Region/TM/tmtype/Data/data.interval`.
16. Finally, the current time –86400 seconds is written to `$TM_SESSION/Region/TM/tmtype/Data/GMT.last` to indicate that *TMthreshold* is finished.

Control

TMthreshold runs in the SEL3 pipeline; no databases, cron jobs, or triggers are used. Control during processing is coordinated using the `GMT.last` file. It terminates automatically when the data chunks have been written.

Interfaces

TMthreshold uses common blocks, including arrays containing the *STA* traces and the threshold traces, to store data internally. The ASCII file `$TM_SESSION/scratch/grid.xyz` is written for use by GMT within *makecdf.2562*, and the GMT *surface* results are stored in `$TM_SESSION/scratch/grid.cdf`. These results are later transferred to the RDF disk loop `$TM_SESSION/Region/TM/tmtype/RDF/grid.rdf`.

Error States

Time intervals are checked extensively to avoid errors. Specific error messages from *TMthreshold* are listed in [\[IDC6.5.14\]](#).

TMmap

TMmap uses the GMT routine *surface* to create maps of the TM results. *TMmap* has a C front end, which uses the *getpar/mstpar* routines; it is usual to use a parameter file to enter the parameters.

▼ Detailed Design

The main subroutine (*d2rproc*) and most of the other subroutines are written in FORTRAN.

Input/Processing/Output

Although it is possible to enter every argument on the command line, it is most convenient to use a parameter file. The par file for *TMmap* may contain the parameters shown in [Table 15](#). Parameters in parameter files may be overridden by placing them after the `par=<parfile>` specification on the command line. The same format is used for the par file and the command line. The search order is from left to right on the command line, with precedence given to parameters found later if there are multiple specifications.

TABLE 15: TM MAP PARAMETERS

| Parameter | Default | Description |
|--------------------------|----------------------------------|---|
| <i>session_directory</i> | none (required) | location of the TM processing environment |
| <i>t1</i> | none (required) | start time of the processing segment |
| <i>t2</i> | none (required) | end time of the processing segment |
| <i>tmtype</i> | Network | name of station subset used for calculating detection thresholds (Network indicates that all stations are used in the processing) |
| <i>scratchdir</i> | <i>session_directory/scratch</i> | name of temporary file storage directory |
| <i>gmtscript</i> | <code>makecdf.2562</code> | name of Bourne shell script doing the actual interpolation and reformatting of the data |
| <i>verbose</i> | none (optional) | if present, diagnostic information will be printed on <code>stdout</code> |

TMmap expects the *tmtype* to be entered as a parameter, rather than reading the `$TM_SESSION/Region/tm_types` file. The default is **Network**. Because **Network** is hard coded in some of the TM scripts, it is best not to attempt to use another *tm_type*.

The detailed processing flow for *DeleteTMStation* is described in the following paragraphs. *DeleteTMStation* writes repeatedly to `$TM_SESSION/Region/TM/tmtype/RDF/GMT.last` in the same way as *CreateTMSession* writes to its own `GMT.last` file.

1. The time interval for the target disk loops is read from `$TM_SESSION/Region/STA/stn/Data/data.interval` with subroutine *dintread*. There may be as many as three attempts to read this file, with 10 second “naps” between attempts. The same subroutine is used to read `$TM_SESSION/Region/STA/stn/RDF/data.interval`, which contains the time interval for the interpolated “surface” (RDF) disk loop. Only one attempt is made to read this file. (If the file is not found, then the data interval in the target disk loops is processed.) One attempt is then made to read `$TM_SESSION/Region/targets`.
2. The sampling interval is extracted from the header of the last target disk loop `$TM_SESSION/Region/TM/tmtype/Data/Target_2562` using *tmcfavail* (in the *tmcf* library).
3. A number of checks are performed on the time intervals, and a decision is made as to whether to continue or not. If continuing, the time interval is divided into chunks, and the data are read from each target disk loop:
 - `$TM_SESSION/Region/TM/tmtype/Data/Target_nnnn`
4. The following RDF files are opened:
 - `$TM_SESSION/Region/TM/tmtype/RDF/grid.rdf` (disk loop)
 - `$TM_SESSION/scratch/grid.xyz` (ASCII file)
5. A call is made to the script *makecdf.2562*, which executes the GMT routine *blockmedian*, writes the temporary file `$TM_SESSION/scratch/grid.xyz.blockmed`, calls the GMT routine *surface*, and then removes the temporary file after *surface* has executed. The output file from *surface* (the last step in *makecdf.2562*) is `$TM_SESSION/scratch/grid.cdf`.
6. Subroutine *readcdf* reads the `grid.cdf` file from *surface*.

▼ Detailed Design

7. The Common Data Format (CDF) data are written to `$TM_SESSION/Region/TM/tmtype/RDF/grid.rdf` after initializing the header (if necessary).
8. Finally, `$TM_SESSION/Region/STA/stn/RDF/data.interval` and `$TM_SESSION/Region/TM/tmtype/RDF/GMT.last` are updated.

Control

TMmap runs in the SEL3 pipeline. No databases, cron jobs, or triggers are used. The Bourne shell script *makecdf.2562* is executed within *TMmap*. *TMmap* uses `GMT.last` to coordinate processing in the same way as *CreateTMSession* uses its `GMT.last` file.

Interfaces

Common blocks are used internally to store data. An ASCII file (`$TM_SESSION/scratch/grid.xyz`) is written for use by GMT within *makecdf.2562*, and the GMT *surface* results are stored in `$TM_SESSION/scratch/grid.cdf`. These are later transferred to the RDF disk loop `$TM_SESSION/Region/TM/tmtype/RDF/grid.rdf`.

Error States

Extensive checks of the time intervals are made to avoid errors. Specific error messages from *TMmap* are listed in [\[IDC6.5.14\]](#).

TMprod

TMprod is a Bourne shell script that generates three sets of plots showing the results of TM for one hour.

Input/Processing/Output

TMprod does not use the *getpar* command line parser, so the arguments to *TMprod* are regular arguments to a shell script and cannot contain a par file. The command line for *TMprod* follows:

```
TMprod <Session> <productdir> <t1> <t2> <database> <bullfile>
```

<Session> is the fully specified path for the TM processing environment (\$TM_SESSION); <productdir> is the fully specified path for files containing hourly results (\$TM_PRODDIR); <t1> is the processing segment start time in epoch format; <t2> is the processing segment end time in epoch format; <database> is the file products database (should be none if no database is used); and <bullfile> is the bulletin filename.

The detailed processing flow for *TMprod* is described as follows:

1. *TMprod* defines some environment variables from \$TM_HOME and internal variables related to the time interval to be processed and files to be written.
2. *TMprod* executes the *tmstatus* script, which calls FORTRAN program *tm_stast*. Subroutine *process_data* reads \$TM_HOME/data/par/tmstatus/tmstatus.dat. *tmstatus.dat*, which is not a standard par file such as one would expect for use with *getpar/mstpar*, but a table of parameters for each station to be included in the status plots. [Table 16](#) describes the parameters.
3. Column 4 of the *tmstatus.par* file contains either a beam name (for single stations) or the name of a file containing a list of beam names (for arrays). The STA beams are read by *tmstacread* (in the *tmcf* library):
 - \$TM_SESSION/Region/STA/stn/Data/TMnnn
4. Results of TM processing are written to \$TM_PRODDIR/yyyy/ddd/status.ddd:hh.mm.data, where yyyy is the year, ddd is the day of year, hh is the hour, and mm the minute. This is an ASCII file containing a table of uptime percentages (number of good samples divided by

▼ Detailed Design

expected number of samples multiplied by 100) and average values of good output samples for each station. The plots themselves are written to `$TM_PRODDIR/yyyy/ddd/status.ddd:hh.mm.ps`.

5. After *tm_stast* is finished, *tmstatus* inserts a row in the **fileproduct** database table before returning to *TMthreshold*.
6. Next, *TMprod* executes the script *plotuptime*.
7. Using the GMT routines *psbasemap*, *pscoast*, and *psxy*, the *uptime* map is written to the following postscript file:

- `$TM_PRODDIR/yyyy/ddd/uptime.ddd:hh.mm.ps`

8. The status data from *tm_stast*: `$TM_PRODDIR/yyyy/ddd/status.ddd:hh.mm.data` are then merged with the following files:
 - `$TM_HOME/data/par/uptime/cur_pri.data`
 - `$TM_HOME/data/par/uptime/offset.data`

psxy is used to plot the color-coded stations (triangles for single stations, circles for arrays) on the plot. Headings for an event bulletin are then written on the plot with *pstext*, but *TMprod* does not give *plotuptime* a bulletin file, so no bulletin information is printed.

9. Before ending, *plotuptime* inserts a row in the **fileproduct** database table and removes the `.gmtcommands` and `.gmtdefaults` files generated automatically by GMT.
10. The final script executed by *TMprod* is *detplot*, which calls the C program *rdf2cdf* twice. This uses the *getpar/mstpar* routines to read two color palette files, `data.cpt.2` and `data.cpt.3`, kept in `staticdir/par/detplot/` ([Table 17](#) shows formats for these files) and converts RDF disk loop data to CDF format. *detplot* reads the binary file `$TM_SESSION/Region/TM/Network/RDF/grid.rdf`.
11. The first time *rdf2cdf* is called, it writes `$TM_SESSION/scratch/detplot/mean.cdf`. The second time, it writes `$TM_SESSION/scratch/detplot/point-max.cdf`. These two files contain the average and worst-case threshold grids, respectively. The files are removed when *detplot* ends.

12. The GMT routines *pscoast*, *grdimage*, *grdcontour*, and *pstext* are used by *detplot* to write the following output postscript file, which contains the average and worst-case threshold maps:
 - `$TM_PRODDIR/yyyy/ddd/detplot.ddd:hh.mm.ps`
13. Before ending, *detplot* inserts a row in the **fileproduct** database table and removes the `.gmtcommands` and `.gmtdefaults` files generated automatically by GMT.

TABLE 16: TMSTATUS.PAR PARAMETERS

| Column and Parameter | Description |
|----------------------|--|
| 1 <i>stn</i> | The station name (for example, ARCES). |
| 2 <i>missteer</i> | The station teleseismic P-wave magnitude correction due to beam-forming signal loss and a log(STA) correction ($sloss/20 + ATcomp$). These numbers are based on tuning studies. For example, 0.09 m_b units must be added to log(STA) from ILAR to make it comparable with log (A/T). In addition, the expected signal loss from beamforming is 1.27 dB for teleseismic events. The total correction for ILAR is 0.16 m_b units. |
| 3 <i>stntype</i> | Provides information on the number of beams used to monitor the teleseismic distance regime. When a single beam is used (as for 3-C stations and small arrays where a single beam covers the teleseismic distances with less than 3 dB signal loss) <i>stntype</i> = single. For larger arrays, several beams are needed to monitor the teleseismic distance regime, and <i>stntype</i> = group. |
| 4 <i>beamfile</i> | The name of the STA trace(s) to be used to represent the noise levels in the teleseismic distance regime. Related to the beam definitions of DFX. For 3-C stations and small arrays (<i>stntype</i> = single), one STA trace calculated from the vertical-component is used. For larger arrays where several beams are used in the teleseismic regime (<i>stntype</i> = group), <i>beamfile</i> is the name of the file containing the beam names. For example, ILAR traces contain STA beams TM017–TM035, and all 19 STA beams are analyzed to provide the noise levels for the status plots. |
| 5 <i>fband</i> | The frequency band used to monitor the teleseismic distance regime (for plotting purposes only). |

▼ Detailed Design

TABLE 17: DATA.CPT PARAMETER FILES

| Line | Description |
|-------------------------------------|---|
| 1 <i>contour interval 1</i> | low contour value, RGB (Red, Green, Blue) values (0–255) of low contour, high contour value, RGB values (0–255) of high contour |
| 2– <i>n</i> ... | red value of low contour (0–255) |
| <i>n</i> <i>contour interval n</i> | low contour value, RGB values (0–255) of low contour, high contour value, RGB values (0–255) of high contour |
| <i>n</i> +1 <i>background color</i> | "B" followed by RGB values (0–255) of background color |
| <i>n</i> +2 <i>foreground color</i> | "F" followed by RGB values (0–255) of foreground color |

Control

TMprod runs when one hour's worth of data are processed in the SEL3 pipeline. It calls other scripts and programs and inserts rows in the **fileproduct** database table. Otherwise it requires no intervention.

Interfaces

Some data are stored in common blocks when *tm_stast* is executed. Whenever data must be transferred between C or FORTRAN programs and Bourne shell scripts, temporary files are written.

Error States

Extensive checks of the time intervals are made to avoid errors. Specific error messages from *TMprod* and the scripts and programs that it calls are listed in [\[IDC6.5.14\]](#).

TMpostreb

TMpostreb is a Bourne shell script that reads the REB and then regenerates the "uptime" maps.

Input/Processing/Output

TMpostreb does not use the *getpar* command line parser so the arguments to *TMpostreb* are regular arguments to a shell script and cannot contain a par file. Exactly four parameters must be entered on the command line:

```
TMpostreb <Session> <productdir> <rebdb> <database>
```

<Session> is the fully specified path for the TM processing environment (\$TM_SESSION); <productdir> is the fully specified path for files containing hourly results (\$TM_PRODDIR); <rebdb> is the name of the database account containing REB information; and <database> is the name of the database account containing the **fileproduct** table with the time for the latest REB entry.

The detailed processing flow for *TMpostreb* is described in the following paragraphs.

1. *TMpostreb* executes the *TMbulletin* script. The purpose of *TMbulletin* is to create event lines from CSS3.0 database tables. *TMbulletin* first checks for the existence of a temporary bulletin file (\$TM_SESSION/Bulletin/TMbulletin.IDCREB.temp). If the file does not already exist, then REB information is read from the database using SQL commands and written to the temporary file. The bulletin covers the interval from *NOW – reb_length* to *NOW*, where *NOW* is the current UTC time from the system clock. The *reb_length* (in seconds) is set in *TMpostreb* and is used in the command line when calling *TMbulletin* (*reb_length*=1209699 is just over two weeks long).
2. *TMbulletin* checks the exit code for the database query. If the code is not zero *TMbulletin* deletes the temporary file and exits; otherwise, the temporary file is renamed \$TM_SESSION/Bulletin/TMbulletin.IDCREB and *TMbulletin* exits. A trap ensures the temporary file is deleted if *TMbulletin* is interrupted.
3. *TMpostreb* checks the exit code and exits if it is nonzero; otherwise, it calls the script *replotuptime*.

▼ Detailed Design

4. *replotuptime* checks for the existence of the log file `$TM_SESSION/scratch/replot.log.last`. If the file does not exist, the start time given to *TMpostreb* is used; otherwise, the start time is replaced by the time in the log file if the log file time is less than the given stop time. 1,800 seconds will be subtracted from the start time. *replotuptime* checks the exit code value and exits if it is nonzero; otherwise, it calls the script *loopuptime*.
5. *loopuptime* executes *plotuptime* (described under *TMpostreb*) for each hour in the specified time interval. *plotuptime* reads the following files:
 - `$TM_PRODDIR/yyyy/ddd/status.ddd:hh.mm.data`
 - `$TM_HOME/data/par/uptime/cur_pri.data`
 - `$TM_HOME/data/par/uptime/offset.data`
 - `$TM_SESSION/Bulletin/TMbulletin.IDCREB`*plotuptime* writes the file:
 - `$TM_PRODDIR/yyyy/ddd/uptime.ddd:hh.mm.ps`*plotuptime* also inserts a row in the **fileproduct** database table and removes the `.gmtcommands` and `.gmtdefaults` files generated automatically by GMT.
6. After *loopuptime* finishes, the exit status is checked again. Then, the last time is written to `$TM_SESSION/scratch/replot.log.last`.

Control

TMpostreb is started by the *rebdone* script. Rows are inserted into the **fileproduct** database table, and some temporary files are written. The exit status of the programs are used to control the behavior of the programs.

Interfaces

TMpostreb uses SQL to read the REB; some temporary files are written.

Error States

The script does not generate many specific error messages of its own. If errors (from *tuxshell* for example) are found in the log files, the operator should check the input parameters, system variables, configuration, and disk space.

TMpostreb calls the Bourne shell scripts *TMbulletin*, which dumps data from the REB to an ASCII file (\$TM_SESSION/Bulletin/TMbulletin.IDCREB) and *replotuptime*, which re-executes *plotuptime* (via script *loopuptime*) for the entire day. *TMbulletin* uses SQL and exits if the status (" \$? " on quitting SQL) is not zero. *TMpostreb*, *loopuptime*, and *replotuptime* also exits at various times if the status is not zero.

LoopCopy

LoopCopy copies all or part of the data in any STA, TM, or RDF disk loop to a new file.

Input/Processing/Output

LoopCopy uses starting and ending times of interest and the size of the new file in seconds as input. These three parameters are not required, and the default values are those found in the old file. If none of these parameters are provided, then the new file is a duplicate of the old file.

LoopCopy is run using the following command line:

```
% LoopCopy par=<parfile> <mode(s)> [update] [verbose[=n]]
```

[Table 18](#) describes the <parfile> parameters used by *LoopCopy*.

▼ Detailed Design

TABLE 18: LOOPCOPY PARAMETERS

| Parameter | Default | Description |
|--------------------------|---------------------------------|---|
| <i>session_directory</i> | none (required) | location of the TM processing environment |
| <i>scratchdir</i> | <Session>/scratch | scratch directory |
| <i>stlist</i> | all stations | comma-separated list of stations |
| <i>t1</i> | starttime of data in input file | starttime of the processing segment |
| <i>t2</i> | endtime of data in input file | endtime of the processing segment |
| <i>staloop</i> | $t2-t1+1$ | size of new STA files in seconds |
| <i>tmloop</i> | $t2-t1+1$ | size of new RDF or TM files in seconds |
| <i>tmlist</i> | all targets | comma-separated list of targets |

LoopCopy expects at least one <mode> parameter. The <mode> parameter should be:

- STA when copying from the STA disk loops in <Session>/Region/STA/<stn>/Data/ for a list of stations
- TM when copying from the TM disk loops in <Session>/Region/TM/<tmttype>/Data/
- RDF for copying from the grid.rdf file in <Session>/Region/TM/<tmttype>/RDF/

Any of these parameters may be present in the same call to *LoopCopy*. For example:

```
% LoopCopy par=<parfile> STA RDF TM
```

copies the files for all desired stations, all desired TM disk loops, and grid.rdf.

Files are read from the appropriate subdirectory in *<Session>* (specified in *<parfile>*), and new files are created in the designated scratch directory. Output filenames have the following formats:

<input_file>.<tmttype> (TM mode, for example, *Target_1001.Network*)

<input_file>.<stn> (STA mode, for example, *TM001.ARCES*)

grid.<tmttype> (RDF mode, for example, *grid.Network*)

Times *t1* and *t2* in the *<parfile>* may be specified either as epoch times or in the format *YYYY-DDD:HH.MM.SS.sss*. Any data in an input file that fall outside the range given by *t1* and *t2* are ignored. Data that fall within this range are copied, and the oldest data are discarded if the new file is too small (due to the *staloop* or *tmloop* given in *<parfile>*).

To change the length of the disk loops, *LoopCopy* overwrites the input files using the *update* option. For example:

```
% LoopCopy par=<parfile> RDF update
```

creates a new RDF file in the designated scratch area and then overwrites the original *grid.rdf* file, destroying the original contents. The file *data.interval* is updated. In the case of STA or TM disk loops, *cf.length* is also updated. The file(s) created in the scratch area are removed.

Control

LoopCopy is not part of normal TM processing. It is a utility that is run from the command line as specified in the previous section.

Error States

LoopCopy has the same (optional) verbosity parameter as *CreateTMSession*. Specific error messages from *LoopCopy* are listed in [\[IDC6.5.14\]](#).

DATABASE DESCRIPTION

Database Design

TM uses a database for describing file products and to obtain REB information. [Figure 11](#) shows the entity-relationship (E-R) model of the schema. Generic file products are described in **fpdescription**, and the rows in this table are linked to the actual product information in **fileproduct** through *typeid*. Three tables are used to extract REB information: **origin**, **originerr**, and **gregion**. The *grn* links the **gregion** table to **origin**, and *orid* links the **origin** and **originerr** tables.

Database Schema

[Table 19](#) shows the usage of database tables by TM. For each table used, the third column shows the purpose for reading or writing each attribute. The relationships between the tables are shown in the E-R diagrams of [Figure 11](#).

TABLE 19: DATABASE USAGE BY TM

| Table | Action | Attribute Usage |
|----------------------|----------------|--|
| fileproduct | writes | <ul style="list-style-type: none"> <i>detplot</i>, <i>plotuptime</i>, and <i>tmstatus</i> insert complete rows as products become available |
| fpdescription | one-time write | <ul style="list-style-type: none"> three complete rows are inserted, one for each product: <i>prodtype</i> = <i>tm_detection</i> <i>prodtype</i> = <i>tm_uptime</i> <i>prodtype</i> = <i>tm_status</i> |
| gregion | reads | <ul style="list-style-type: none"> <i>grname</i> is used by <i>TMbulletin</i> for REB information |
| originerr | reads | <ul style="list-style-type: none"> <i>smajax</i>, <i>sminax</i>, <i>strike</i>, <i>sdobs</i>, <i>szz</i>, and <i>stt</i> are used by <i>TMbulletin</i> for REB information |
| origin | reads | <ul style="list-style-type: none"> <i>orid</i>, <i>jdate</i>, <i>time</i>, <i>lat</i>, <i>lon</i>, <i>depth</i>, <i>mb</i>, <i>ml</i>, <i>ms</i>, <i>ndef</i>, <i>etype</i> and <i>auth</i> are used by <i>TMbulletin</i> for REB information |

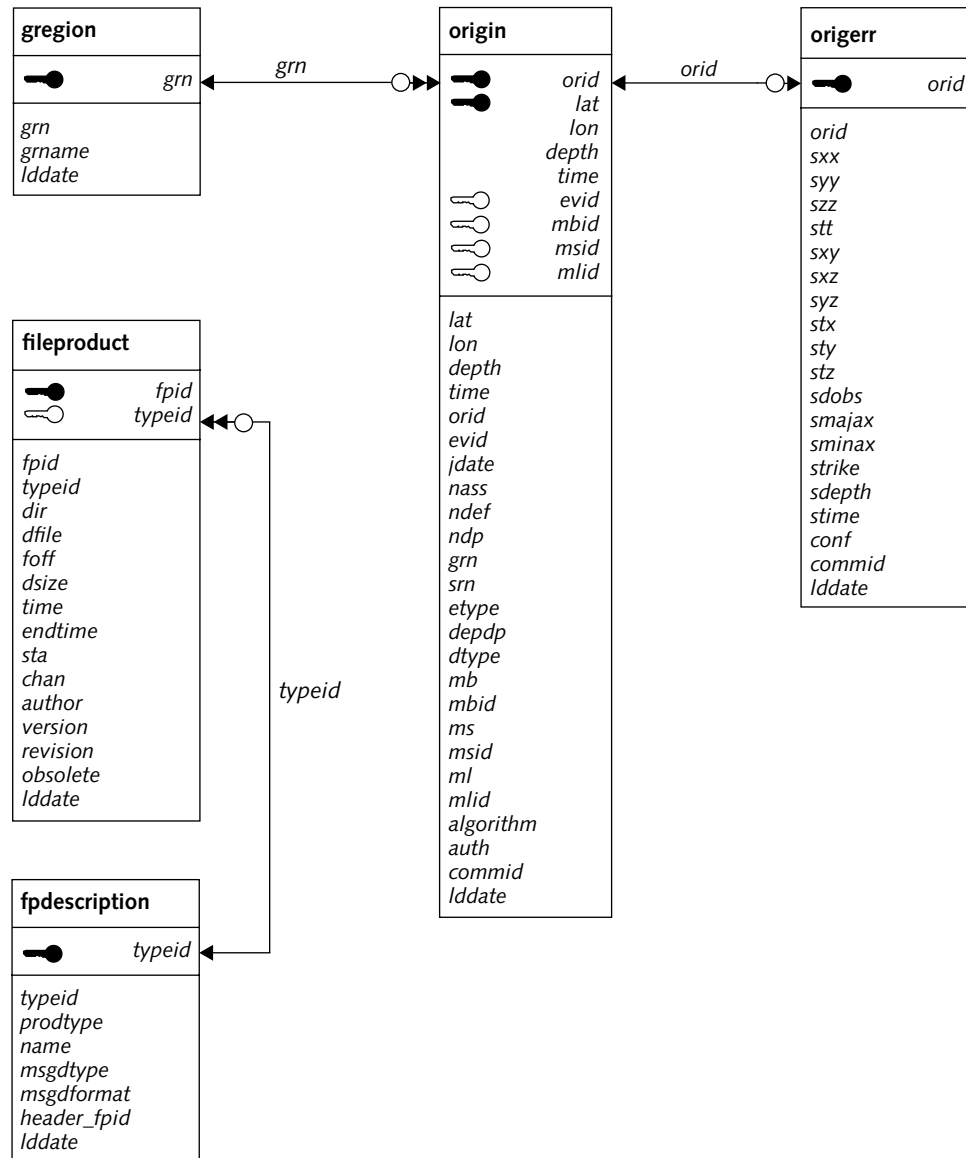


FIGURE 11. TM TABLE RELATIONSHIPS

References

The following sources supplement or are referenced in document:

- [DOD94a] Department of Defense, "Software Design Description," *Military Standard Software Development and Documentation*, MIL-STD-498, 1994.
- [Gan79] Gane, C., and Sarson, T., *Structured Systems Analysis: Tools and Techniques*, Prentice-Hall, Inc., Englewood Cliffs, NJ, 1979.
- [Han85] Hannon, W., "Seismic Verification of a Comprehensive Test Ban," *Science* 227, pp. 251–257, 1985.
- [Har85] Harjes, H.P., "Global Seismic Network Assessment for Teleseismic Detection of Underground Nuclear Explosions," *Journal of Geophysics*, Volume 57, pp. 1–13, 1985.
- [IDC5.1.1Rev2] Science Applications International Corporation, Veridian Pacific-Sierra Research, *Database Schema, Revision 2*, SAIC-00/3057, PSR-00/TN2830, 2000.
- [IDC5.1.3Rev0.1] Science Applications International Corporation, Veridian Pacific-Sierra Research, *Configuration of PIDC Databases*, SAIC-01/3022, PSR-99/TN1114, 2001.
- [IDC6.5.2Rev0.1] Science Applications International Corporation, *Distributed Application Control System (DACs) Software User Manual, Revision 0.1*, SAIC-00/3038, 2000.
- [IDC6.5.14] Science Applications International Corporation, *Threshold Monitoring Software Users Manual*, SAIC-01/3004, 2001.

▼ References

- [IDC6.5.18] Science Applications International Corporation, *Continuous Data Subsystem CD-1.1 Software User Manual*, SAIC-01/3006, 2001.
- [IDC7.3.1] Science Applications International Corporation, *Distributed Application Control System (DACs)*, SAIC-01/3001, 2001.
- [IDC7.4.1] Science Applications International Corporation, *Continuous Data Subsystem*, SAIC-99/3006, 1999.
- [Ken91b] Kennett, B., and Engdahl, E., "Traveltimes for Global Earthquake Location and Phase Identification," *Geophysical Journal International*, Volume 105, pp. 429–465, 1991.
- [Kvæ92] Kværna, T., "Initial Results From Global Generalized Beamforming," *Semiannual Technical Summary, 1 April – 30 September 1992*, NORSAR Scientific Report 1-92/93, NORSAR, Kjeller, Norway, 1992.
- [Kvæ94] Kværna, T., Ringdal, F., Iversen, H., and Larsen, N. H. K., "A System for Continuous Seismic Threshold Monitoring," *Final Report, Semiannual Technical Summary, 1 April– 30 September 1994*, NORSAR Scientific Report 1-94/95, NORSAR, Kjeller, Norway, 1994.
- [Kvæ96] Kværna, T., "Tuning of Processing Parameters for Global Threshold Monitoring at the IDC," *Semiannual Technical Summary, 1 April – 30 September 1996*, NORSAR Scientific Report 1-96/97, NORSAR, Kjeller, Norway, 1996.
- [Kvæ97a] Kværna, T., "Threshold Magnitudes," *Semiannual Technical Summary, 1 October 1996 – 31 March 1997*, NORSAR Scientific Report, 2-96/97, NORSAR, Kjeller, Norway, 1997.
- [Kvæ97b] Kværna, T., Ringdal, F., Bassdshaug, U., and Iversen, H., "Status of the Global Threshold Monitoring (TM) System," *Semiannual Technical Summary, 1 April – 30 September 1997*, NORSAR Scientific Report, 1-97/98, NORSAR, Kjeller, Norway, 1997.

- [Rin79] Ringdal, F. and Fyen, J., "Analysis of Global P-wave Attenuation Characteristics Using ISC Data Files," *Semiannual Technical Summary, 1 April – 30 September 1979*, NORSAR Scientific Report 1-79/80, NORSAR, Kjeller, Norway, 1979.
- [Rin86] Ringdal, F., "Study of Magnitudes, Seismicity and Earthquake Detectability Using a Global Network," *Bulletin of the Seismological Society of America*, Volume 76, pp. 1641–1659, 1986.
- [Rin89] Ringdal, F., and Kværna, T., "A Multi-channel Processing Approach to Real-time Network Detection, Phase Association, and Threshold Monitoring," *Bulletin of the Seismological Society of America*, Volume 79, pp. 780–798, 1989.
- [Rin92] Ringdal, F. and Kværna, T., "Continuous Seismic Threshold Monitoring," *Geophysical Journal International*, Volume 111, pp. 505–514, 1992.
- [Ser89] Sereno, T. J. and Bratt, S. R., "Seismic Detection Capability at NORESS and Implications for the Detection Threshold of a Hypothetical Network in the Soviet Union," *Journal of Geophysical Research*, Volume 94, pp. 10397–10414, 1989.
- [Syk82] Sykes, L. and Evernden, J., "The Verification of a Comprehensive Nuclear Test Ban," *Scientific American*, Volume 247, pp. 47–55, 1982.
- [Vei72] Veith, K. F., and Clawson, G. E., "Magnitude of Short-period P Wave Data," *Bulletin of the Seismological Society of America*, Volume 62, pp. 435–453, 1972.
- [Vin92] Vinje, V., Iversen, E., Gjølstdal, H., and Åstebøl, K., "Traveltime and Amplitude Estimation Using Wavefront Construction," *Abstract of paper presented at the 54th Meeting and Technical Exhibition of the European Association of Exploration Geophysicists in Paris, France, 1–5 June 1992*.
- [Wah96] Wahl, D. D., *User's Manual for the Detection and Feature Extraction Program*, Science Applications International Corporation, SAIC-96/1098, 1996.

▼ References

- [Wes95] Wessel, P. and Smith, H. F., "New version of Generic Mapping Tools Released," *EOS Transactions, American Geophysical Union*, Volume 76, pp. 329, 1995.

Glossary

Symbols

3-C

Three-component.

A

amplitude

Zero-to-peak height of a waveform in nanometers.

array

Collection of sensors distributed over a finite area (usually in a cross, triangle, or concentric pattern) and referred to as a single station.

arrival

Detected signal that has been associated to an event. First, the Global Association (GA) software associates the detection to an event. Later, during interactive processing, many arrivals are confirmed, improved, or added by visual inspection.

ASCII

American Standard Code for Information Interchange. Standard, unformatted 256-character set of letters and numbers.

azimuth

Direction, in degrees clockwise with respect to North, from a station to an event.

B

background noise

Natural movements of the earth, oceans, and atmosphere as seen on S/H/I sensors (usually measured in data preceding signals).

beam

(1) Waveform created from array station elements that are sequentially summed after being steered to the direction of a specified azimuth and slowness. (2) Any derived waveform (for example, a filtered waveform).

beamform

Sum a set of waveforms from array station elements with time delays introduced to compensate for the time it takes a wave to travel across the array.

▼ Glossary

bulletin

Chronological listing of event origins spanning an interval of time. Often, the specification of each origin or event is accompanied by the event's arrivals and sometimes with the event's waveforms.

C**channel**

Component of motion or distinct stream of data.

CMR

Center for Monitoring Research.

coda

Signal of a given phase, which follows the initial waveform of that phase.

command

Expression that can be input to a computer system to initiate an action or affect the execution of a computer program.

component

(1) One dimension of a three-dimensional signal; (2) The vertically or horizontally oriented (north or east) sensor of a station used to measure the dimension; (3) One of the parts of a system; also referred to as a module or unit.

Comprehensive Nuclear-Test-Ban Treaty Organization

Treaty User group that consists of the Conference of States Parties, the Executive Council, and the Technical Secretariat.

Computer Software Component

Functionally or logically distinct part of a computer software configuration item; possibly an aggregate of two or more software units.

Computer Software Configuration Item

Aggregation of software that is designated for configuration management and treated as a single entity in the configuration management process.

configuration

(1) (hardware) Arrangement of a computer system or components as defined by the number, nature, and interconnection of its parts. (2) (software) Set of adjustable parameters, usually stored in files, which control the behavior of applications at run time.

COTS

Commercial-Off-the-Shelf; terminology that designates products such as hardware or software that can be acquired from existing inventory and used without modification.

CSC

See [Computer Software Component](#).

CSCI

See [Computer Software Configuration Item](#).

CSS 3.0

Center for Seismic Studies (CSS) version 3 database schema, including a format for storing time-series data in disk files and database descriptors of that data.

CTBT

Comprehensive Nuclear-Test-Ban Treaty (the Treaty).

CTBTO

See [Comprehensive Nuclear-Test-Ban Treaty Organization](#).

D**DACS**

Distributed Application Control System. This software supports inter-application message passing and process management.

dB

Decibel.

DBA

Database Administrator.

deg.

Degrees (as a distance).

detection

Probable signal that has been automatically detected by the *Detection and Feature Extraction (DFX)* software.

DFX

Detection and Feature Extraction. *DFX* is a programming environment that executes applications written in Scheme (known as *DFX* applications).

disk loop

Storage device that continuously stores new waveform data while simultaneously deleting the oldest data on the device.

E**element**

Single station or substation of a sensor array, referred to by its element name (such as YKR8), as opposed to its array name (YKA in this example). (2) Data storage location in a data array.

entity-relationship (E-R) diagram

Diagram that depicts a set of entities and the logical relationships among them.

envelope

Overall waveform immediately following a major phase, usually P or S.

event

Unique source of seismic, hydroacoustic, or infrasonic wave energy that is limited in both time and space.

F**fileproduct**

Database table whose records describe files containing products.

▼ Glossary

filesystem

Named structure containing files in sub-directories. For example, UNIX can support many filesystems; each has a unique name and can be attached (or mounted) anywhere in the existing file structure.

G**GB**

Gigabyte. A measure of computer memory or disk space that is equal to 1,024 megabytes.

H**Hz**

Hertz.

I**IASPEI**

International Association of Seismology and Physics of the Earth's Interior.

IDC

International Data Centre.

IMS

International Monitoring System.

IPC

Interprocess communication. The messaging system by which applications communicate with each other through *libipc* common library functions. See [tuxshell](#).

J**Julian date**

Increasing count of the number of days since an arbitrary starting date.

K**km**

Kilometer.

L**LAN**

Local Area Network.

M**magnitude**

Empirical measure of the size of an event (usually made on a log scale).

MB

Megabyte. 1,024 kilobytes.

N**network**

Spatially distributed collection of seismic, hydroacoustic, or infrasonic stations for which the station spacing is much larger than a wavelength.

noise

Incoherent natural or artificial perturbations of the waveform trace caused by ice, animals migrations, cultural activity,

equipment malfunctions or interruption of satellite communication, or ambient background movements.

O

ORACLE

Vendor of the database management system used at the PIDC and IDC.

P

P phase

Seismic wave that travels from the event to the station as a compressional wave through the solid earth.

par

See [parameter](#).

parameter

User-specified token that controls some aspect of an application (for example, database name, threshold value). Most parameters are specified using [*token* = *value*] strings, for example, `dbname=mydata/base@oracle`.

parameter (par) file

ASCII file containing values for parameters of a program. Par files are used to replace command line arguments. The files are formatted as a list of [*token* = *value*] strings.

period

Average duration of one cycle of a phase, in seconds per cycle.

phase

Arrival that is identified based on its path through the earth.

PIDC

Prototype International Data Centre.

pipeline

1) Flow of data at the IDC from the receipt of communications to the final automated processed data before analyst review. 2) Sequence of IDC processes controlled by the DACS that either produce a specific product (such as a Standard Event List) or perform a general task (such as station processing).

primary seismic

IMS seismic station(s) or data that is (are) part of the detection network.

probability of detection

Probability estimate that an arrival from a given event will be detected at a station given the location and magnitude of the event, the average noise level and its standard deviation at the station, and the signal-to-noise detection threshold.

R

REB

Reviewed Event Bulletin; the bulletin formed of all S/H/I events that have passed analyst inspection and quality assurance review.

▼ Glossary

S**s**

Second(s) (time).

script

Small executable program, written with UNIX and other related commands, that does not need to be compiled.

SEL3

Standard Event List 3; S/H/I bulletin created by totally automatic analysis of both continuous data and segments of data specifically down-loaded from stations of the auxiliary seismic network. Typically, the list runs 12 hours behind real time.

S/H/I

Seismic, hydroacoustic, and infrasonic.

slowness

Inverse of velocity, in seconds/degree; a large slowness has a low velocity.

snr

Signal-to-noise ratio.

Solaris

Name of the operating system used on Sun Microsystems hardware.

SQL

Structured Query Language; a language for manipulating data in a relational database.

STA (or STAV)

Short-term average. A running average of the absolute value or squared value of a waveform. The averaging window is short in duration compared to the LTA.

station

Collection of one or more monitoring instruments. Stations can have either one sensor location (for example, BGCA) or a spatially distributed array of sensors (for example, ASAR).

steer

Construct a beam using time delays consistent with a particular azimuth and slowness.

T**teleseismic**

1) (distance) Source-to-seismometer separations of 20 degrees or more. (2) (event) Recorded at distances where the first P and S waves from shallow events have traveled paths through the mantle/core.

theoretical arrival

Point where an arrival is expected to appear on a waveform, based on an event's location and depth.

TM

Threshold monitoring. A technique to keep track of the minimum detectable event based on noise levels at stations.

Treaty

Comprehensive Nuclear-Test-Ban Treaty (CTBT).

Tuxedo

Transactions for UNIX Extended for Distributed Operations.

tuxshell

Process in the Distributed Processing CSCI used to execute and manage applications. See [IPC](#).

U**UNIX**

Trade name of the operating system used by the Sun workstations.

UTC

Universal Coordinated Time.

V**version**

Initial release or re-release of a computer software component.

W**waveform**

Time-domain signal data from a sensor (the voltage output) where the voltage has been converted to a digital count (which is monotonic with the amplitude of the stimulus to which the sensor responds).

wavenumber

Vector, k , in the direction of a propagating wave whose magnitude is given by the inverse wavelength of the wave scaled by a factor of 2π .

Web

World Wide Web; a graphics-intensive environment running on top of the Internet.

Index

A

AddTMStation [5](#), [6](#), [21](#), [26](#)
 control [46](#)
 error states [47](#)
 input/processing/output [46](#)
 attenuation curves [39](#)
AutoDRM [29](#)
 azimuth tolerances [15](#)

B

beambasic parameters [37](#)
 beamforming signal loss [19](#)
 beam mis-steering signal loss [20](#)
 Beams/TMnnn parameters [42](#)

C

CopyPSFile [5](#), [22](#)
CreateTMSession [5](#), [6](#), [21](#), [25](#)
 control [45](#)
 error states [45](#)
 input [34](#)
 output [40](#)
ctms [6](#), [22](#)

D

DACS [29](#)
 data.cpt parameter files [58](#)
 data flow [33](#)
DeleteTMStation [5](#), [6](#), [21](#), [26](#)
 control [47](#)
 error states [47](#)
 input/processing/output [47](#)
 detection capability traces [12](#)
 for a global grid system [13](#)
detplot [5](#), [22](#), [27](#), [41](#)
DFX [iii](#), [24](#), [25](#), [26](#), [32](#)

F

F77 [22](#)
fileproduct [23](#), [27](#), [59](#), [60](#), [64](#), [65](#)
fpdescription [23](#), [28](#), [64](#), [65](#)
 rows for TM products [24](#)

G

GMT [vi](#)
gmt [22](#)
gregion [27](#), [64](#), [65](#)
grn [64](#)

I

IASP91 [vi](#)
interval [23](#), [24](#), [64](#)

▼ Index

L

LoopCopy [5](#), [6](#), [21](#), [61](#), [62](#)
 control [63](#)
 error states [63](#)
 input/processing/output [61](#)
loopuptime [5](#), [22](#), [61](#)

M

M77 [22](#)
magnitude calibration [15](#)
makecdf.2562 [5](#), [22](#), [26](#), [41](#)
makelibs [5](#)
makemods [5](#)
missteer.Dlow-Dhig parameters [38](#)

N

netcdf [22](#)
ngbase [22](#)
ngut [22](#)

O

orid [64](#)
origerr [27](#), [65](#)
origin [27](#), [64](#), [65](#)
originerr [64](#)

P

plotuptime [5](#), [27](#), [41](#), [61](#)
prefilter cutoffs [18](#)

R

rdf2cdf [5](#), [21](#), [27](#)

REB [27](#), [34](#)
rebdone [23](#), [29](#), [34](#), [60](#)
Region/TM/TM_recipes/ [43](#)
replotuptime [5](#), [22](#), [61](#)

S

schema [23](#)
Session directory structure [40](#)
site [38](#)
slowness tolerances [15](#)
STA envelopes [19](#)
station-specific parameters [17](#)
sunmath [22](#)
surface [26](#)

T

tab.PTM parameters [39](#)
Target_XXXX parameters [43](#)
targets.n parameters [36](#)
threshold traces [10](#)
time tolerances [15](#)
tm_beambasic [5](#), [21](#)
tm_globrec [5](#), [21](#)
tm_stast [5](#), [21](#), [27](#)
TMbulletin [5](#), [22](#), [27](#), [61](#)
tmcf [22](#)
TMmap [5](#), [6](#), [21](#), [32](#), [54](#)
 control [54](#)
 detailed processing flow [53](#)
 error states [54](#)
 input/processing/output [52](#)
 interfaces [54](#)
 parameters [52](#)
TMpostreb [5](#), [22](#), [28](#), [34](#), [61](#)
 command line [59](#)
 control [60](#)
 detailed processing flow [59](#)
 error states [61](#)
 input/processing/output [59](#)

- interfaces [60](#)
- TMprod* [5](#), [22](#), [27](#), [32](#), [55](#), [58](#)
 - command line [55](#)
 - control [58](#)
 - detailed processing flow [55](#)
 - error states [58](#)
 - input/processing/output [55](#)
 - interfaces [58](#)
- TM.site parameters [38](#)
- tmstatus* [5](#), [22](#), [27](#), [41](#)
- tmstatus.par* parameters [57](#)
- TMthreshold* [5](#), [21](#), [32](#)
 - control [51](#)
 - detailed processing flow [49](#)
 - error states [51](#)
 - input/processing/output [48](#)
 - interfaces [51](#)
 - parameters [48](#)
- travel-time table [39](#)
- trvminmax* parameters [45](#)
- tuxshell* [iii](#), [61](#)
- typeid* [64](#)

